Customizable HSB Sensor Interfaces on Lattice Avant-X Platforms

Reference Design

FPGA-RD-02329-1.4

July 2026

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.

Abbreviations	Definition
APB	Advanced Peripheral Bus
AOC	Active Optical Cable
AXI	Advanced eXtensible Interface
AXIS	AXI4-Stream
AWB	Automatic White Balance
BOOTP	Bootstrap Protocol
CDC	Clock Domain Crossing
CRC	Cyclic Redundancy Check
CMOS	Complementary Metal-Oxide-Semiconductor
CUDA	Compute Unified Device Architecture
DCT	Discrete Cosine Transform
DDR	Dual Data Rate
DDRDLL	Dual Data Rate Delay-Locked Loop
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
Hex	Hexadecimal
HIF	Host Interface
HSB	Holoscan Sensor Bridge
I2C	Inter-Integrated Circuit
ICMP	Internet Control Message Protocol
IP	Intellectual Property (Core)
ISP	Image Signal Processing
LSCC	Lattice Semiconductor Company
LUT	Look-Up Table
MAC	Media Access Control
MIPI	Mobile Industry Processor Interface
MJPEG	Motion JPEG (video encoded as a sequence of JPEG images)
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
PMA	Physical Medium Attachment
PROM	Programmable Read-Only Memory
QSFP28	Quad Small Form-factor Pluggable 28
RD	Reference Design
RTL	Register-Transfer level
SDK	Software Development Kit

Abbreviations	Definition
SERDES	Serializer/Deserializer
SFP	Small Form-factor Pluggable
SPI	Serial Peripheral Interface
STA	Static Timing Analysis
UDP	User Datagram Protocol

1. Introduction

This user guide describes the reference design for customizable HSB sensor interfaces on Lattice Avant™-X platforms. The supported platform includes the Avant-X Versa Board. This board acts as the Holoscan Sensor Bridge (HSB) between sensors and the NVIDIA host. The customizable interfaces in this reference design are divided into two interface categories.

- **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.

Note: On the Avant-X Versa Board, one Ethernet port and one camera connector are available for testing.

The primary objective of the reference design is to establish a robust and reliable bridge between the HSB interfaces and the NVIDIA host. The reference design leverages the NVIDIA 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 Avant-X Versa Board functions as the Holoscan Sensor Bridge solution.
   - Performs sensor data pre-processing to reduce computational load on the host system.
   - The reference design targets the Lattice Avant-X device (LAV-AT-X70 speed grade 3).
     The key components of the reference design include:
     - Sensor Interface
       - Supports MIPI CSI-2 camera sensors that are 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 10GbE or 25GbE speeds.
       - Enables low-latency and high-bandwidth communication.
3. Host (NVIDIA Jetson AGX Orin/Thor 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.

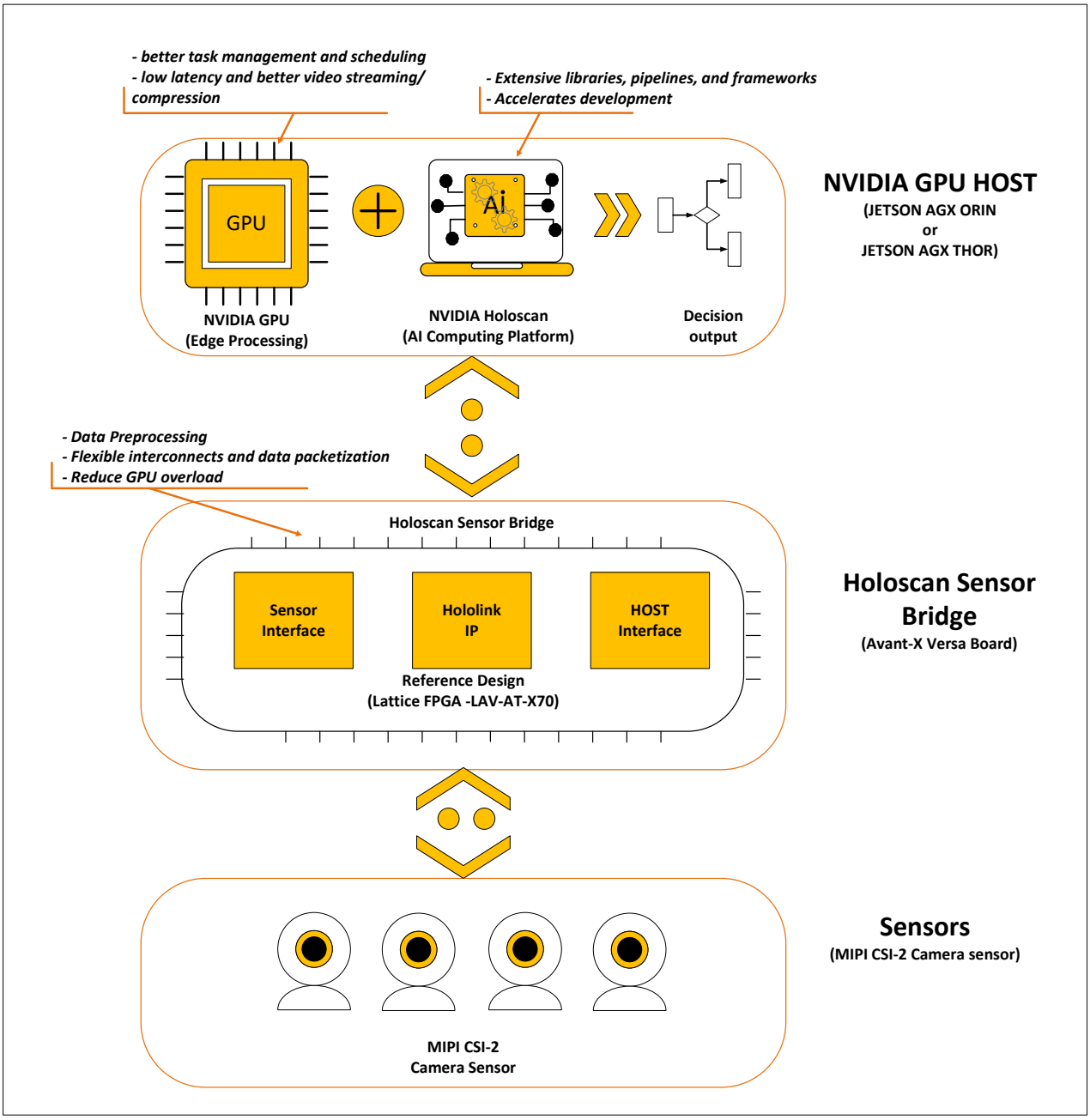


Figure 1.1. Overview of Customizable HSB Sensor Interface RD with NVIDIA ecosystem

1.1. Reference Design: Quick Facts

1.1.1. Avant-X Versa Board with Jetson AGX Orin/Thor

Table 1.1. Summary of Avant-X Versa Board with NVIDIA Jetson AGX Orin

General	Target Devices	Avant-X FPGA (LAV-AT-X70-3LFG1156C)
	Source code format	Verilog and SystemVerilog
Simulation	Functional simulation	N/A
	Timing simulation	N/A
	Testbench	N/A
	Testbench format	N/A
Software Requirements	Software tool and version	Lattice Radiant™ software version 2025.2.1.321.0
	IP version (if applicable)	• CSI-2/DSI D-PHY RX IP v2.0.0 • OSC IP v2.1.0 • PLL IP v2.6.1 • Ethernet 10GbE IP core v3.3.1 (MAC + PHY) • Image Signal Processing (ISP) Lattice Semiconductor Reference Design IP v1.1.1 • Launch LSCC ISP Builder Linux Orin v1.1.0 • MJPEG Lattice Semiconductor Reference Design IP • Hololink IP v2511
	Host Software (AGX Orin)	• JetPack 6.2.1 • NVIDIA Holoscan SDK 3.7.0 • NVIDIA Jetson Linux 36.4.4 • HSB SDK 2.5.0
Hardware Requirements	Board	• Avant-X Versa Board • IMX258 camera module • Ethernet 10GBase-T (Cat 6 RJ-45) SFP+ module (10G Ethernet) • Display Monitor
	Host	• NVIDIA Jetson AGX Orin Developer Kit
	Cable	• DisplayPort cable • Ethernet Cat 6 cable

Table 1.2. Summary of Avant-X Versa Board with NVIDIA Jetson AGX Thor

General	Target Devices	Avant-X FPGA (LAV-AT-X70-3LFG1156C)
	Source code format	Verilog and SystemVerilog
Simulation	Functional simulation	N/A
	Timing simulation	N/A
	Testbench	N/A
	Testbench format	N/A
Software Requirements	Software tool and version	Lattice Radiant software version 2025.2.1.321.0
	IP version (if applicable)	• CSI-2/DSI D-PHY RX IP v2.1.0 • OSC IP v2.1.0 • PLL IP v2.6.1 • Ethernet 10GbE IP v3.3.1 (MAC + PHY) • Ethernet 25GbE IP v2.2.0 (MAC + PHY) • Image Signal Processing (ISP) Lattice Semiconductor Reference Design IP v1.1.1 • LSCC ISP Builder Linux Thor v1.1.0 • MJPEG Lattice Semiconductor Reference Design IP • Hololink IP v2511

	Host Software (AGX Thor)	• JetPack 7.0 • NVIDIA Holoscan SDK 3.7.0 • NVIDIA Jetson Linux 38.2 • HSB SDK 2.5.0
Hardware Requirements	Board	• Avant-X Versa Board • IMX258 camera module • Display Monitor
	Host	• NVIDIA Jetson AGX Thor Developer Kit
	Cable	• DisplayPort cable • 100G QSFP28 to 4×25G SFP28 Breakout Active Optical Cable (AOC)

1.2. Quick-Start Guide

IMX258 Camera Streaming Setup, refer to the Implementing the Reference Design section

This section provides instructions for setting up and streaming video from the IMX258 camera module to the NVIDIA Jetson AGX Orin or Thor developer kit over a 10GbE or 25GbE Ethernet connection.

Jetson AGX Orin Developer Kit Setup

This section covers the steps for setting up the NVIDIA Jetson AGX Orin developer kit

Jetson AGX Thor Developer Kit Setup

This section covers the steps for setting up the NVIDIA Jetson AGX Thor developer kit

1.3. 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 D-PHY interface.
  - Enables integration of various camera types.
  - In testing, the Avant-X Versa Board is used with a single MIPI CSI-2 camera.
  - Performs data pre-processing, supporting raw MIPI CSI-2 data capture as well as optional image processing such as ISP-based image enhancement and MJPEG image decoding.
- 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 Hololink IP 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 10GbE or 25GbE.
  - 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 Avant-X Versa Board is used with a single Ethernet port configured for either 10GbE or 25GbE.

Features are enabled and implemented according to the targeted platform configured as Holoscan Sensor Bridge. For detailed instructions, see the Customizing the Reference Design section. You can tune the reference design to meet your own platform requirements.

1.4. Naming Conventions

1.4.1. Nomenclature

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

1.4.2. Signal Names

- _*n* are active low (asserted when value is logic 0)
- _*i* are input signals
- _*o* are output signals

2. Directory Structure and File Overview

Figure 2.1 illustrates the directory structure of the reference design.

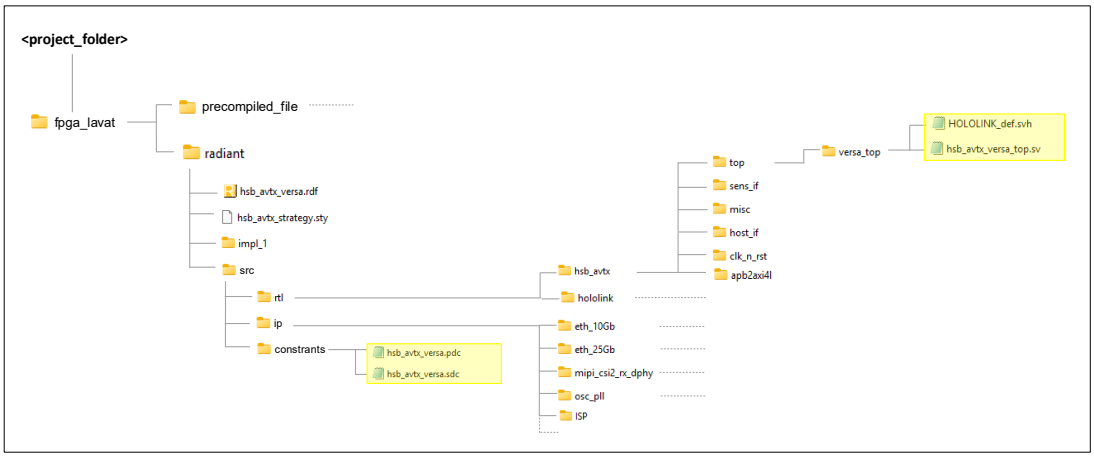


Figure 2.1. Directory Structure

Table 2.1. Folder Structure Description

Reference Design Specific	Folder/File	Description
Common Folders/files	radiant/src/rtl/hsb_avtx/	This folder contains HDL code for the sensor interface, host interface, top-level design, and other components.
	radiant/src/rtl/hololink/	This folder contains the encrypted Hololink IP. The current version is v2511.
	radiant/ip/	This folder contains IP cores used in the reference design, such as the oscillator, PLLs, MIPI CSI-2 RX D-PHY IP, Ethernet 10GbE IP (MAC + PHY), and Ethernet 25GbE IP (MAC + PHY).
	radiant/ip/ISP/	This folder contains ISP and MJPEG Lattice Semiconductor Reference design IP.
	radiant/hsb_avtx_strategy.sty	This file is the strategy file used for both reference designs. Custom settings are applied to improve place-and-route results.
	radiant/imp_1	This folder contains reports and logs generated during FPGA bitstream compilation.
	precompiled_file /	The folder consists of the FPGA compiled bitstream for the reference design.
Avant-X Versa Board	radiant/hsb_avtx_versa.rdf	This file is the Lattice Radiant software project file for the Avant-X Versa reference design.
	radiant/src/constraints/hsb_avtx_versa.sdc	The SDC file for Avant-X Versa board is used to configure the intended Ethernet physical pins.
	radiant/src/constraints/hsb_avtx_versa.pdc	This file contains the PDC file for the Avant-X Versa Board.
	radiant/src/rtl/hsb_avtx/top/versa_top/hsb_avtx_versa_top.sv	This file is the Avant-X Versa reference design top file.
	radiant/src/rtl/hsb_avtx/top/versa_top/HOLOLINK_def.svh	This file contains define switches for the Avant-X Versa reference design.

Note: When customizing your reference design, you must modify the files shown in Figure 2.1.

- radiant/src/rtl/hsb_avtx/top/versa_top/HOLOLINK_def.svh
- radiant/src/rtl/hsb_avtx/top/versa_top/hsb_avtx_versa_top.sv
- radiant/src/constraints/hsb_avtx_versa.sdc
- radiant/src/constraints/hsb_avtx_versa.pdc

3. Functional Description

3.1. NVIDIA Host Developer Kits

This section provides an overview of the NVIDIA host developer kits, highlighting key features of the Jetson AGX Orin and Jetson AGX Thor developer kits. For additional information, refer to the NVIDIA Developer webpage and search for Jetson Developer Kits.

3.1.1. Jetson AGX Orin Developer Kit

Figure 3.1. Jetson AGX ORIN Developer Kit

The NVIDIA Jetson AGX Orin is a compact, high-performance edge AI platform for workloads that require substantial on-device AI processing. It delivers 200–275 INT8 TOPS of compute through an Ampere-architecture GPU with 1,792–2,048 CUDA cores and 56–64 Tensor Cores. The system integrates either an 8-core or 12-core Arm Cortex-A78AE CPU running up to 2.2 GHz and 32 GB or 64 GB of LPDDR5 memory with 204.8 GB/s bandwidth. These resources enable efficient execution of complex AI pipelines and high-throughput data streams.

Jetson AGX Orin supports the NVIDIA software stack, including Isaac for robotics, DeepStream for vision AI, and Riva for conversational AI, allowing you to build applications across multiple AI domains.

Jetson AGX Orin provides a single Multi-Gigabit Ethernet (MGBE) interface. Depending on the carrier board and PHY implementation, the interface typically supports 1GbE, 2.5GbE, 5GbE, and 10GbE Ethernet. This bandwidth is sufficient for receiving UDP-based sensor data from a Holoscan Sensor Bridge, although it operates at lower throughput than NVIDIA Thor's 25GbE Ethernet interfaces. In the current reference design, only the 1GbE and 10GbE Ethernet link speeds are enabled.

Jetson AGX Orin supports Precision Time Protocol (PTP) through its integrated MGBE interface, enabling accurate time synchronization with attached sensors. The platform fully supports UDP transport, providing low-latency, real-time data streaming. Although its Ethernet bandwidth is lower than that of NVIDIA Thor, Orin can accommodate multiple sensor inputs as long as the combined throughput remains within the capacity of the MGBE interface.

3.1.2. Jetson AGX Thor Developer Kit

Figure 3.2. Jetson AGX Thor Developer Kit

The NVIDIA Jetson AGX Thor is a next-generation robotics and physical-AI computing platform built for highly demanding real-time sensor workloads. It provides up to 2,070 FP4 TFLOPS of AI compute powered by the NVIDIA Blackwell GPU architecture, offering more than 7.5× greater AI performance and substantially improved energy efficiency compared to Jetson AGX Orin.

The system integrates 128 GB of LPDDR5X memory with 273 GB/s of bandwidth, enabling efficient handling of large-scale multimodal data streams, including high-resolution camera, LiDAR, and radar inputs. To meet real-time robotics requirements, Jetson AGX Thor includes a 14-core Arm Neoverse-V3AE CPU that provides deterministic compute performance for sensor fusion, perception, and rapid decision-making. When combined with the NVIDIA Holoscan framework, the platform can execute complex sensor pipelines with sub-10 ms latency, supporting advanced autonomous-system and robotics workloads.

The Jetson AGX Thor provides high-bandwidth Ethernet connectivity for integration with the Holoscan Sensor Bridge. The system includes a QSFP28 interface supporting four 25 Gbps Ethernet links, delivering a combined throughput of 100 Gbps. This capacity allows the host to ingest multiple high-rate UDP sensor streams from imaging and ranging devices without performance bottlenecks. Thor also incorporates a 5GbE RJ45 Ethernet port that can be used for system control, secondary data paths, or redundancy, depending on system requirements.

All high-speed Ethernet interfaces on Jetson AGX Thor support IEEE-1588 Precision Time Protocol (PTP), enabling accurate alignment of sensor timestamps when used with the Holoscan Sensor Bridge. This capability is essential for synchronized streaming across multiple sensors, including cameras, LiDAR, radar, and microphones. Thor also provides full UDP transport support, offering low-latency, non-blocking data delivery suitable for real-time systems. With its 25GbE Ethernet links, the platform can maintain continuous high-throughput UDP streams, ensuring reliable frame-level sensor data reception even under heavy system load.

3.2. Lattice-Powered Holoscan Sensor Bridge Platform

This section provides an overview of the targeted Lattice-powered Holoscan Sensor Bridge platform. The supported Avant-X platform includes the Avant-X Versa Board.

3.2.1. Avant-X Versa Board

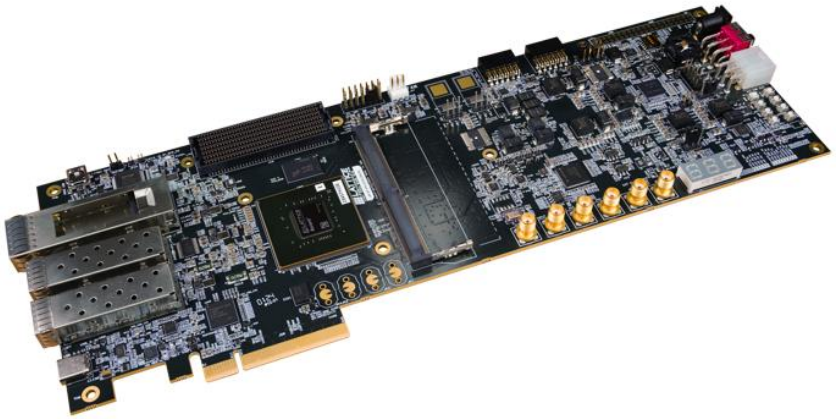


Figure 3.3. Avant-X Versa Board

The Lattice Avant-X Versa Board is the FPGA platform used to implement the Holoscan Sensor Bridge for this application. It features the Avant-X FPGA, a mid-range device designed for high-throughput sensor processing, including 25 Gbps SERDES, advanced security features (AES-256-GCM, ECC512/RSA4096 authentication, and SHA-3), and extensive I/O connectivity, including FMC+, PMOD, SMA, and high-speed Ethernet interfaces.

As part of the NVIDIA Holoscan ecosystem, the Avant-X Versa Board implements the FPGA-based Holoscan Sensor Bridge, which packetizes high-speed sensor data such as MIPI CSI-2 camera streams and transmits the data through 10GbE or 25GbE to the host platform with ultra-low latency. This usage aligns with the Lattice Customizable HSB Sensor Interfaces reference design, which supports both the CertusPro™-NX and Avant-X Versa boards for MIPI CSI-2-to-Ethernet bridging within the Holoscan workflows.

Integrating the Avant-X device into the Holoscan pipeline enables flexible, real-time sensor ingestion, allowing you to efficiently stream and process sensor data directly on NVIDIA Jetson AGX Orin or Thor platforms. It's high memory bandwidth (supporting LPDDR4/DDR5 up to 2,400 Mbps) and validated 25GbE Ethernet IP cores help ensure robust performance for demanding AI-driven sensing systems. For more details about the Avant-X Versa Board, refer to the Avant-G/X Versa Board User Guide (FPGA-EB-02063).

3.3. Hololink IP and Customizable Interfaces

3.3.1. Overview

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. The block diagram of the Holoscan Sensor Bridge FPGA IP is illustrated in Figure 3.4. The Sensor Interface and Host (Ethernet) Interface blocks are implemented using FPGA vendor-specific logic. In this reference design, Lattice Semiconductor is responsible for integrating and maintaining the IP for both the Sensor Interface and Host Interface components.

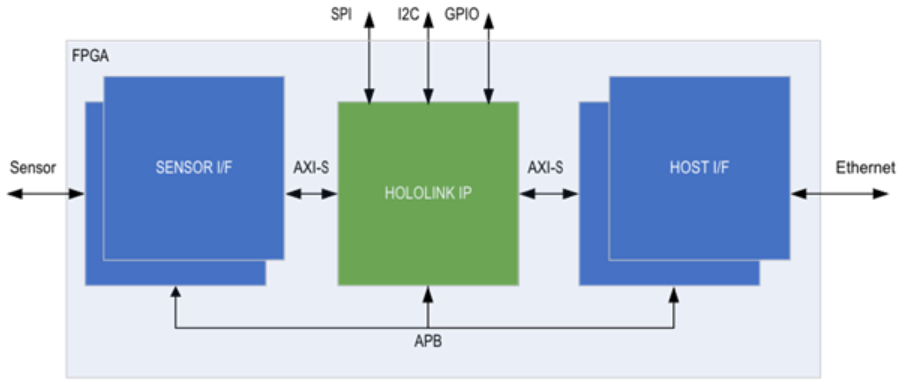


Figure 3.4. Holoscan Sensor Bridge IP in FPGA

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.

A detailed explanation of the NVIDIA Hololink IP is provided in the NVIDIA documentation webpage. Search for *Holoscan Sensor Bridge* and then select the respective HSB SDK release. Review the document for further understanding. This section provides only an overview of the IP.

3.3.2. Sensor Interface

3.3.2.1. MIPI CSI-2 RX D-PHY 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 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 sensor lane numbers and lane rates.

MIPI CSI-2 Payload Data to AXI-Stream Payload

Figure 3.5 shows the block diagram of the MIPI CSI-2 sensor interface, which outputs MIPI CSI-2 RAW data, MJPEG bitstream, or ISP YUV422 data format.

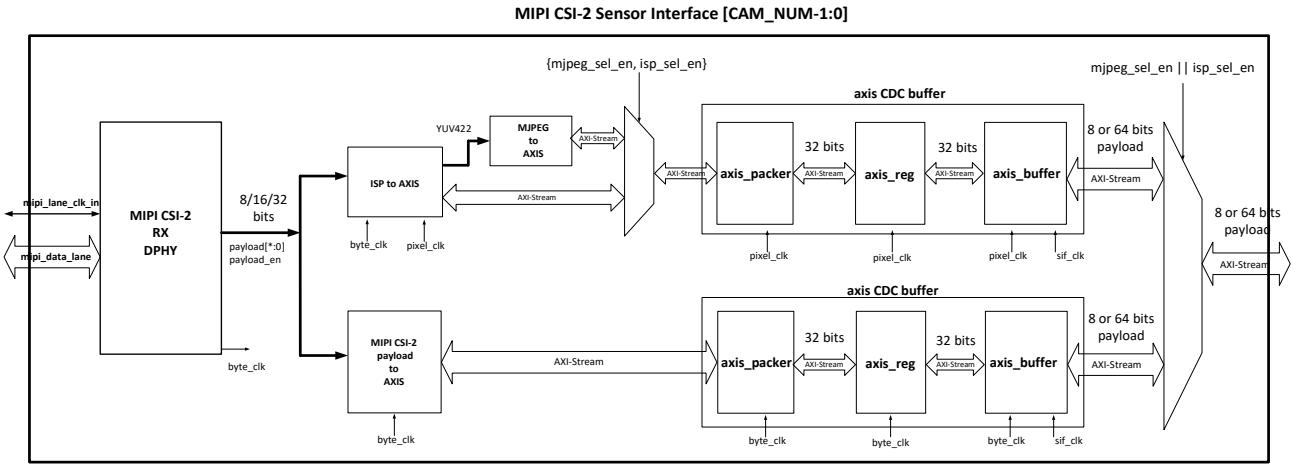


Figure 3.5. Overview block diagram of MIPI CSI-2 Sensor Interface

The MIPI CSI-2 sensor interface includes five primary components: *MIPI CSI-2 RX D-PHY instance*, *MIPI CSI-2 RAW10 to AXIS*, *ISP to AXIS*, *MJPEG to AXIS*, and *AXIS CDC buffer*. The number of MIPI CSI-2 sensor interfaces varies based on the assigned number of cameras.

1. **MIPI CSI-2 RX D-PHY Instance**

   The MIPI CSI-2 RX D-PHY instance instantiates MIPI CSI-2 RX D-PHY IPs that serve as the primary interface for MIPI CSI-2 image sensors. The reference design targets the LAV-AT-X70 device package; therefore, only the soft MIPI CSI-2/DSI RX D-PHY IP is supported. This configurable IP block receives MIPI CSI-2 data streams and converts them into 8-bit, 16-bit, or 32-bit parallel data, depending on the lane configuration selected during IP generation.

   The IP allows you to set the MIPI CSI-2 lane rate based on the requirements of the camera sensor. It also generates an output byte clock derived from the data type and forwards it to the AXIS blocks. If the sensor provides a continuous D-PHY clock, you should disable the RX_FIFO and drive clk_byte_fr_i directly using clk_byte_hs_o.

   Compared to the hardened version, the soft MIPI CSI-2 RX D-PHY IP has performance limitations that vary by FPGA family and package. For detailed specifications and configuration guidance, see the CSI-2/DSI D-PHY RX IP Core User Guide (FPGA-IPUG-02081).

   The MIPI CSI-2 RX D-PHY instance supports the PRESET MIPI configuration method, which means the MIPI CSI-2 RX D-PHY IP is pre-generated before FPGA image compilation. The number of MIPI CSI-2 RX D-PHY IPs that are pre-generated is listed in the Generating the IP section.

2. **MIPI CSI-2 RAW10 to AXIS**

   The MIPI CSI-2 RAW10 payload data is converted into an AXI4-Stream (AXIS) interface. The module extracts image data packets from the CSI-2 stream and reformats them into AXIS-compliant payloads for downstream processing. This conversion supports parallel RAW10 output in 8-bit, 16-bit, or 32-bit formats, as supported by the MIPI CSI-2 RX D-PHY IP.

3. **ISP to AXIS Instance**

   The ISP to AXIS instance consists of two main functional components:

   a. Byte-to-Pixel Conversion module

   b. ISP IP module

The *Byte-to-Pixel Conversion module* performs clock-domain crossing between the MIPI CSI-2 byte clock domain and the pixel clock domain. Incoming MIPI CSI-2 RAW10 payload data is unpacked and converted into a pixel-formatted data stream suitable for ISP processing. This pixel-formatted data is then processed by the image processing functions enabled within the ISP IP. All ISP processing runs in the pixel clock domain, with the pixel clock set to 160 MHz. The current Byte-to-Pixel conversion module supports only MJPEG operation; the supported resolution is to 1920 × 1080.

The *ISP IP* allows you to select an ISP processing configuration that best suits your application requirements. Three processing domains are supported within the ISP pipeline: Bayer, RGB, and YUV. These domains form a sequential processing pipeline, not parallel paths. Color-space conversion stages (Debayer, RGB-to-YUV, and YUV444-to-YUV422) determine the transition points between processing domains. Regardless of the selected ISP functions, the final output of the ISP IP is always YUV422 for Holoscan application.

The current version of the ISP IP supports a limited set of ISP algorithms and processing steps in three color spaces including Bayer, RGB, and YUV. The supported ISP steps are detailed in the following tables.

Table 3.1. Supported Bayer processing and converter steps

Bayer Processing	Descriptions	Available Configurations
Black Level Correction (BLC)	Correct per-plane black offset on Bayer data.	Per-channel black level offsets.
Digital Gain (DG)	Apply gain on Bayer domain.	Per-channel gains.
Image Statistics	Collect image statistics (passthrough step).	Output Min/Max/Avg, bright/dark count thresholds.
Bayer converter	**Descriptions**	**Available Configurations**
Debayer (Linear 3x3)	Convert Bayer to RGB.	None

Table 3.2. Supported RGB processing and converter steps

RGB Processing	Descriptions	Available Configurations
Black Level Correction (BLC)	Apply RGB-domain black-level correction.	Per-channel offsets.
Digital Gain (DG)	Apply RGB gain.	R/G/B gain, per-channel mode.
AWB (Gray World)	White-balance correction using the gray world algorithm	None
Color Correction Matrix (CCM)	Color correction using 3x3 transform.	Matrix coefficients.
1-Dimensional Look-Up-Table (LUT1D)	Tone and gamma curve remapping.	Curve type and curve parameters via LUT dialog.
RGB converter	**Descriptions**	**Available Configurations**
RGB2YUV	Convert RGB to YUV444.	None

Table 3.3. Supported YUV444 processing and converter steps

YUV444 Processing	Descriptions	Available Configurations
Chroma Saturation	Adjust U/V saturation strength.	Saturation gain.
Sharpen	Luma sharpening with unsharp mask.	Gaussian window size, sharpen amount, threshold.
YUV444 converter	**Descriptions**	**Available Configurations**
YUV444toYUV422	Convert YUV444 to YUV422.	None

The output payload format of the ISP IP is fixed to YUV422. Based on the MUX control signal, the YUV422 payload is either routed to the MJPEG IP or forwarded to the Hololink IP after being converted into an AXI-Stream (AXIS)-compliant payload.

4. MJPEG to AXIS

When the MJPEG MUX control signal is selected, the YUV422 payload output from the ISP instance is routed to the MJPEG IP for compression. The MJPEG IP encodes the incoming video stream as a sequence of independent JPEG frames. Each frame is processed using the standard JPEG compression pipeline, including DCT, quantization, and Huffman encoding. However, unlike conventional JPEG mode, the Huffman table is not embedded in each frame header; instead, the encoder assumes the use of the standard JPEG Huffman table internally.

By excluding the Huffman table from every frame header, MJPEG mode reduces header size and minimizes overhead, making it well suited for continuous video streaming applications such as camera interfaces and real-time data transmission. Despite this optimization, the encoding process and output format remain JPEG-compliant, with each frame properly terminated by an End-of-Image (EOI) marker. This approach is particularly efficient in systems where the decoder already has knowledge of the standard Huffman tables and does not require them to be transmitted repeatedly.

The resulting MJPEG bitstream is output after being formatted into an AXI-Stream (AXIS) compliant interface. The MJPEG IP operates with a pixel clock of 160 MHz and supports 1920 × 1080 (Full HD) resolution. When a four-lane MIPI sensor is used, the MIPI lane rate must not exceed 400 Mbps. This MJPEG IP is a Lattice Semiconductor proprietary reference design provided in encrypted form and is integrated as part of the Holoscan reference design.

5. AXIS CDC Buffer

- 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 helps maintain data integrity and prevents downstream modules from processing corrupted or incomplete payloads.
- 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, which helps manage timing, pipeline depth, and synchronization between modules.
- AXIS Buffer Module
  The AXIS Buffer module receives AXI4-Stream data from the AXIS Reg module and transfers it through a dual-clock FIFO (DC_FIFO) to support clock domain crossing (CDC). This enables reliable data movement between components that operate in separate clock domains.
  Key functions include:
  - Data buffering: Temporarily stores AXIS data and control signals to manage timing differences.
  - Clock-domain crossing: Uses a DC_FIFO to safely transfer data between asynchronous clock domains.
  - Flow control: Maintains AXIS protocol integrity using TVALID, TREADY, and TLAST signals across domains.

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

3.3.3. Host Interface through Ethernet Link

The Avant-X device provides high-speed Ethernet connectivity using integrated IP cores and external transceiver modules. This section describes the available 10GbE and 25GbE Ethernet configurations, along with their key components, integration methods, and functional behavior.

3.3.3.1. Ethernet 10GbE Support with 10GBase-T SFP+ Transceiver

The Avant-X device supports 10GbE 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 using AXI or Avalon, depending on the FPGA architecture.
- PHY IP Core (PCS + PMA)
  Handles low-level physical layer functions and interfaces directly with the SFP+ module.
  Features:
  - Encoding/Decoding: Supports 64b/66b schemes.
  - Scrambling/Descrambling: Maintains signal integrity.
  - Link Synchronization: Establishes and maintains a stable link.
  - High-speed serial PMA interface to the SFP+ module
  - Error Handling: Optional Forward Error Correction (FEC) for enhanced reliability.
- 10GBase-T SFP+ Transceiver Module
  A compact, hot-swappable module for 10GbE Ethernet connectivity.
  Features:
  - Hot-Swappable: Enables module replacement without shutting down the system.
  - Media Support: Compatible with copper and fiber (for example, 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 PHY 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.

3.3.3.2. Ethernet 25GbE Support with 25GBASE-SR SFP28 Transceiver

The Avant-X device supports 25GbE Ethernet using 25GBASE-SR SFP28 transceivers. This setup uses the Lattice 25GbE Ethernet MAC+PHY IP core, providing high-speed 25 Gbps connectivity. The integrated MAC and PHY communicate through a 25-Gigabit Media-Independent Interface (25GMII), enabling seamless data transmission and reception between the host system and Ethernet networks in 25GBASE-R applications.

Key Components:

- MAC IP Core
  Manages Ethernet frame processing, enforces media-access rules, supports flow control (pause frames, PFC), CRC/FCS checks, and collects statistics. Operates on a 128-bit data path at 195.3125 MHz with AXI4-Stream TX/RX interfaces.
- PHY IP Core
  Handles PCS (64b/66b encoding) and PMA for the serial interface, with lane merging, loopback modes, and AXI4-Lite management.
- 25GBase-SR SFP28 Transceiver Module
  A compact, hot-swappable module used for 25GbE connections.
  Features:
  - Hot-swappable: Enables module replacement without shutting down the system.
  - Media support: Compatible with fiber (for example, SR for short range, LR for long range).
  - Diagnostics: Supports real-time monitoring of temperature, power, and signal quality.

The 25GbE Ethernet MAC+PHY IP core integrates with the SFP28 module to process and transmit Ethernet frames, providing high-speed, low-latency 25 Gbps communication for advanced networking applications.

3.4. Multi Camera Frame Synchronization

Multiple camera frame synchronization is achieved through a common synchronized signal generated from the FPGA. Camera sensors and modules from different vendors typically allow an external triggerable signal, FSIN/FSYNC/XVS/XTRIG, to be connected to start capturing the video frame. Synchronizing cameras allows two or more cameras to achieve stereo vision and makes depth computation possible.

In the Holoscan Sensor Bridge FPGA, a module named vsync_gen is provided to generate an IEEE 1588 PTP time-stamping-based signal source to be connected to the external trigger signal of the camera to achieve synchronization.

The PTP Time Stamping synchronized signal allows multiple cameras in the Holoscan Sensor Bridge environment to achieve sub-microsecond synchronization.

The vsync_gen module is clocked by ptp_clk in the system; hence, all the counters run at a resolution of approximately 10 ns.

Below are the parameters and configurations allowable in the vsync_gen modules, programmable through HSB SDK.

- PERIOD (10, 30, 60, 90, 120 Hz)
- PULSE EXPOSURE (pulse width) programmable at a resolution of 10 ns
- DELAY START programmable at a resolution of 10 ns
- START VALUE (polarity), active high or active low

3.5. Dual-boot Feature

Dual-boot allows an FPGA to store and load multiple configuration bitstreams (images) from an external SPI flash. This capability provides a reliable fallback mechanism, ensuring the system remains operational even if a primary or alternate configuration becomes corrupted. In a typical deployment, the FPGA contains a primary bitstream (default boot image) and a golden bitstream (safe recovery image).

Key functional behaviors are described below:

- Normal boot flow
  Upon power-up, the device attempts to load the primary bitstream. If loading fails, the FPGA automatically falls back to the golden bitstream, ensuring a valid configuration is always active.
- Fallback and robustness
  If the FPGA fails to load an alternate bitstream, such as when a bitstream is corrupted, it automatically falls back to the golden bitstream.
- External flash address mapping
  Each bitstream image is assigned to a fixed starting address in SPI flash. These addresses must match those defined in the design (example boot_addr parameters) and in the Radiant Deployment Tool configuration.

For more details on the multi-boot feature of the Lattice Avant platform, refer to the Lattice Avant Multi-Boot User Guide (FPGA-TN-02314).

3.6. Clocking and Reset Scheme

3.6.1. Avant-X Versa Board

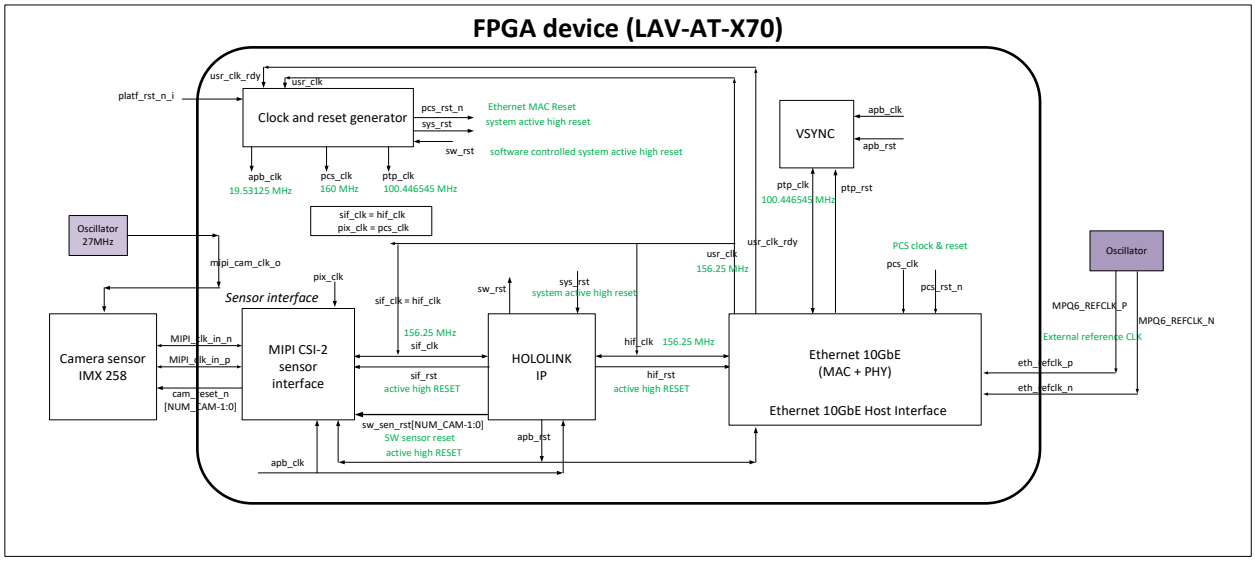


Figure 3.6. Reset and Clock Domain Block Diagram for Ethernet 10GbE

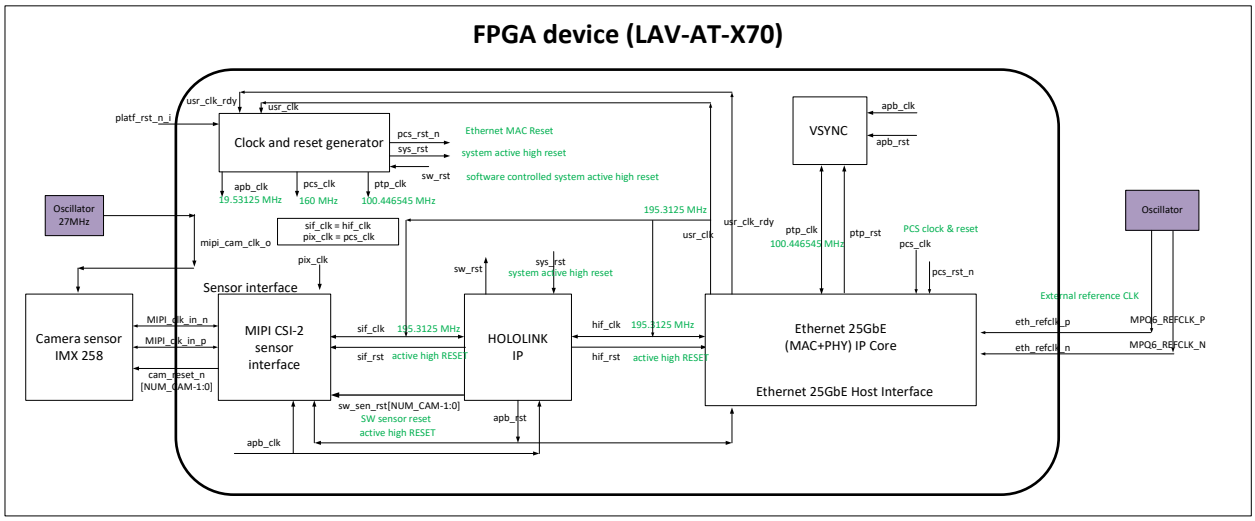


Figure 3.7. Reset and Clock Domain Block Diagram for Ethernet 25GbE

The following tables list the clock frequencies and their distribution.

Table 3.4. Clock Domain Distribution for Reference Design with Ethernet 10GbE

Clocks	Frequency (MHz)	Description
usr_clk	322.266	PCS output clock is used as a reference clock for the PLL to generate system clocks (such as apb_clk and hif_clk).
pcs_clk	160	PCS calibration clock.
OSC_IN_125MHz	125	Platform clock input used for debug hook signals.
eth_refclk_p	156.25	External Ethernet reference clock.
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 the sensor interface.
apb_clk	19.5313	APB bus clock.
ptp_clk	100.45 MHz	Precision time protocol clock.

Table 3.5. Clock Domain Distribution for Reference Design with Ethernet 25GbE

Clocks	Frequency (MHz)	Description
usr_clk	195.3125	Ethernet IP TX output clock used as a reference clock for the PLL to generate system clocks (such as apb_clk).
pcs_clk	160	PCS calibration clock.
eth_refclk_p	156.25	External Ethernet reference clock.
hif_clk	195.3125	Directly assigned from usr_clk and is used for host AXIS interface.
sif_clk	195.3125	Directly assigned from hif_clk and is used as the AXIS clock for sensor interface.
apb_clk	19.5313	APB bus clock.
ptp_clk	100.45 MHz	Precision time protocol clock.

Table 3.6. Reset Distribution for 10GbE and 25GbE Ethernet

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.
ptp_rst	Precision time protocol reset.

4. Customizing the Reference Design

4.1. Customization Steps for the Reference Design

This section provides a detailed walkthrough of the steps required to configure and to verify the customized reference design. Figure 4.1 illustrates the steps involved to enable customization and support for various customer-specific configurations. However, details on reference design compilation and implementation are described in the Compiling the Reference Design section and Implementing the Reference Design sections.

The screenshots provided in this section are for reference only. The details may vary depending on the version of the IP or software being used.

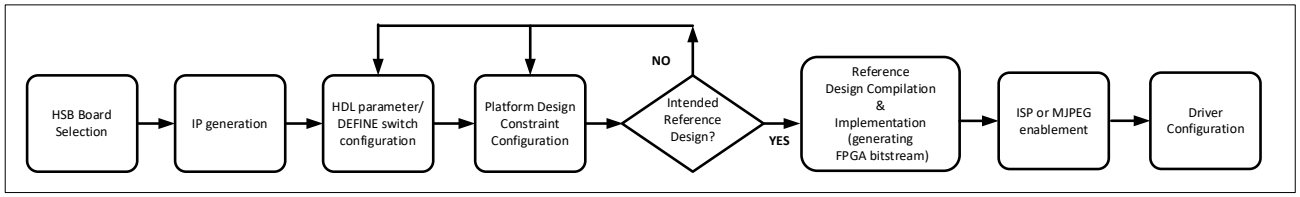


Figure 4.1. Steps Flow to Customize and Verify the Reference Design

4.1.1. Selecting the Holoscan Bridge Board

Customization of the reference design depends on the target hardware platform. The design can be implemented on Lattice-provided platforms, such as the Avant-X Versa Board or on a user-designed custom platform. In this reference design, several HDL parameters are preset for specific platforms to simplify integration; these mappings are described in later sections. If you are developing a custom platform, you can define additional parameters to meet your system requirements.

4.1.2. Generating the IP

4.1.2.1. MIPI CSI-2 RX D-PHY IP generation

The reference design enables the MIPI CSI-2 RX D-PHY IP to interface with MIPI CSI-2 camera sensors. The reference design uses the Preset configuration method.

Preset Configuration Method

The MIPI CSI-2 RX D-PHY IP is generated based on the specifications of the camera sensor being used. The key parameters to configure include:

- Number of data lanes
- MIPI lane rate
- Gear Ratio (fixed to 8)
- MIPI synchronizer clock frequency (fixed to 160 MHz)

These parameters must be correctly set before generating the IP to ensure compatibility with the camera sensor. The gear ratio for the Avant-X family device is fixed at 8, and the MIPI synchronizer clock frequency in the reference design is set to 160 MHz. The reference design also provides preconfigured MIPI CSI-2 RX D-PHY IPs for you to choose from. Table 4.1 shows the available IP configuration options that you can select. The IP is selected based on user settings, as described in the HDL Parameter and DEFINE Switch Configuration section.

Table 4.1. MIPI CSI-2 RX D-PHY 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
1,440	4 lanes	mipi_csi2_rx_DPHY_1440Mbps_4L.ipx
1,500	4 lanes	mipi_csi2_rx_DPHY_1500Mbps_4L.ipx

4.1.2.2. Byte-to-Pixel Conversion IP

The Byte-to-Pixel conversion IP is a Lattice Semiconductor reference design IP provided in encrypted form and integrated as part of the reference design. The IP is located under the ./radiant/src/ip/ISP/b2p_raw10raw8/ directory.

This module performs clock-domain crossing between the MIPI CSI-2 byte clock domain and the pixel clock domain. Incoming MIPI CSI-2 RAW10 payload data is unpacked and converted into a pixel-formatted data stream suitable for ISP processing.

Limitations: The current implementation supports MJPEG operation only; at a fixed resolution of 1920 × 1080.

4.1.2.3. ISP Reference Design IP

The ISP IP is a Lattice Semiconductor proprietary reference design IP provided in encrypted form and integrated as part of the reference design. The IP is located in the radiant/src/ip/ISP/mISP/ directory.

Specifications:

- Input format: Pixel-formatted data stream at 1920 × 1080 (Full HD).
- Output format: YUV422 or YUV422 with AXI-Stream-compliant data.
- Clock domain: Pixel clock, fixed at 160 MHz.

The ISP IP sub-functionality is user-configurable. To optimize image quality for your specific environment, tune the ISP configuration and regenerate the ISP IP. Figure 4.2 illustrates the steps involved in developing and deploying the ISP reference design IP in a Holoscan application.

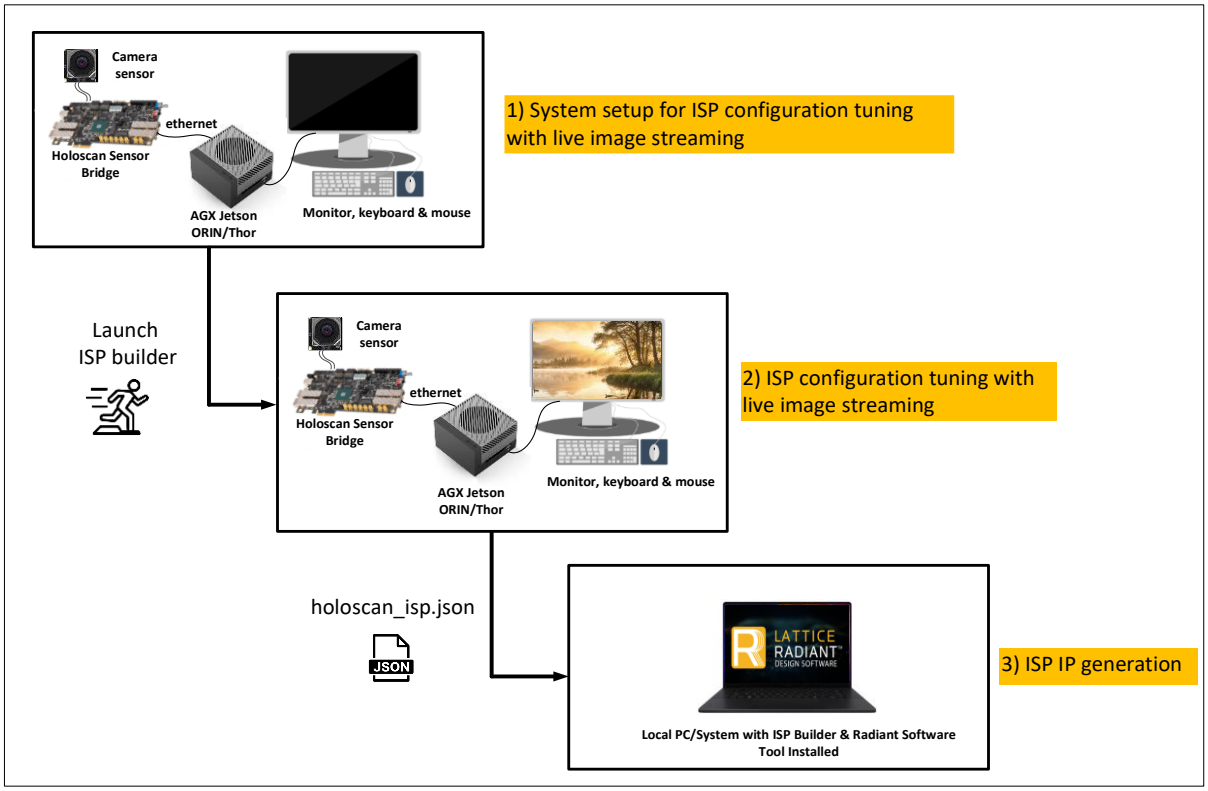


Figure 4.2. Steps to Generate ISP Lattice Semiconductor Reference Design IP

4.1.2.3.1. System Setup for ISP Configuration Tuning with Live Image Streaming

Before proceeding, ensure the following prerequisites are met:

- The system must first be prepared for Holoscan system testing using the Avant-X Versa Board platform with NVIDIA Jetson AGX Orin or Jetson AGX Thor Host. Follow the procedure in the Implementing the Reference Design section of this document.
- The generated FPGA bitstream includes support for the ISP hardware module. You can use demo FPGA bitstream which is provided in the Avant-X Customizable HSB Sensor Interfaces Reference Design – Source Code.

Note: Verify that the MIPI-CSI2 image streams successfully on the Jetson AGX Orin or Thor host before proceeding.

In addition to the system setup for ISP configuration tuning, the LSCC ISP Builder Linux Orin or LSCC ISP Builder Linux Thor software is also required. Copy the tool to the NVIDIA Jetson AGX Orin or Thor Host. These tools are included with the reference design. The software is tested on NVIDIA Jetson AGX Orin or Thor Host, which runs on Linux OS. The software tool (lscc_isp_builder_linux_orin) is located in the radiant/src/ip/ISP/mISP/lscc_isp_builder/ directory of the Customizable HSB Sensor Interface reference design.

4.1.2.3.2. ISP Configuration Tuning with Live Image Streaming

Step 1. Launch LSCC ISP Builder Linux Orin or LSCC ISP Builder Linux Thor software

By default, the reference design includes a pre-generated ISP IP with JSON configuration file. Use this file as a baseline configuration to improve image streaming quality. The JSON file is in the directory shown Figure 4.3 and described in Table 4.2.

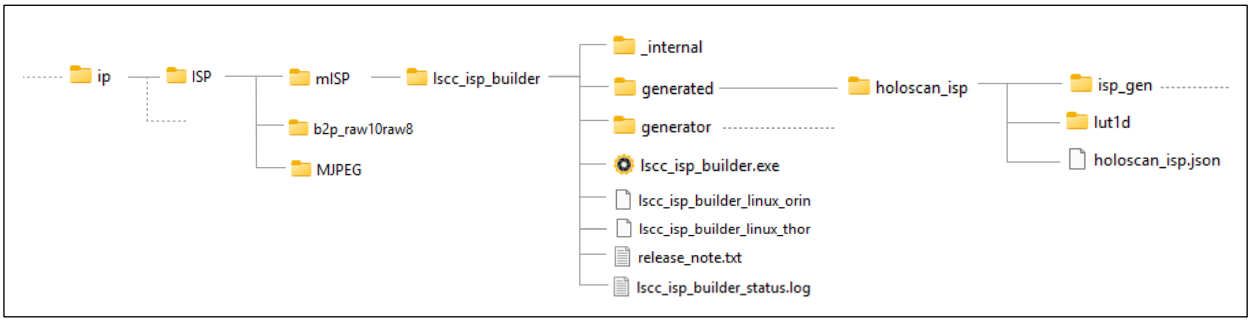


Figure 4.3.Generated ISP IP Folder Structure

Table 4.2. Generated ISP IP Folder Structure Description

Folder/File	Description
lscc_isp_builder/release_note.txt	Release note about the ISP builder includes version number and quick start steps to execute ISP builder.
lscc_isp_builder/lscc_isp_builder_linux_orin	Linux executable file used to tune ISP configuration with live image streaming on NVIDIA Jetson AGX Orin Host.
lscc_isp_builder/lscc_isp_builder_linux_thor	Linux executable file used to tune ISP configuration with live image streaming NVIDIA Jetson AGX Thor Host.
lscc_isp_builder/lscc_isp_builder.exe	Executable file used to generate ISP IP using JSON config file as input.
lscc_isp_builder/generator/	Folder consists of the ISP IP generator related files .
lscc_isp_builder/generated/holoscan_isp/isp_gen	The path that defined by user when generating the ISP IP. This folder consists of the generated ISP IP files.
lscc_isp_builder/generated/holoscan_isp/holoscan_isp.json	This JSON file is generated based on the JSON configuration file used to generate the ISP IP.

Run the LSCC ISP Builder Linux Orin or LSCC ISP Builder Linux Thor software on the host. When the GUI launches, change the camera streaming option to Network Stream to enable ISP live stream tuning. Load the pre-generated JSON configuration file as the initial configuration (see Step 1 in Figure 4.4).

Input/Output Specification:

- Input: Bayer-patterned RAW10 data
- Output: YUV422 format
- Target resolution: 1920 × 1080 (Full HD)

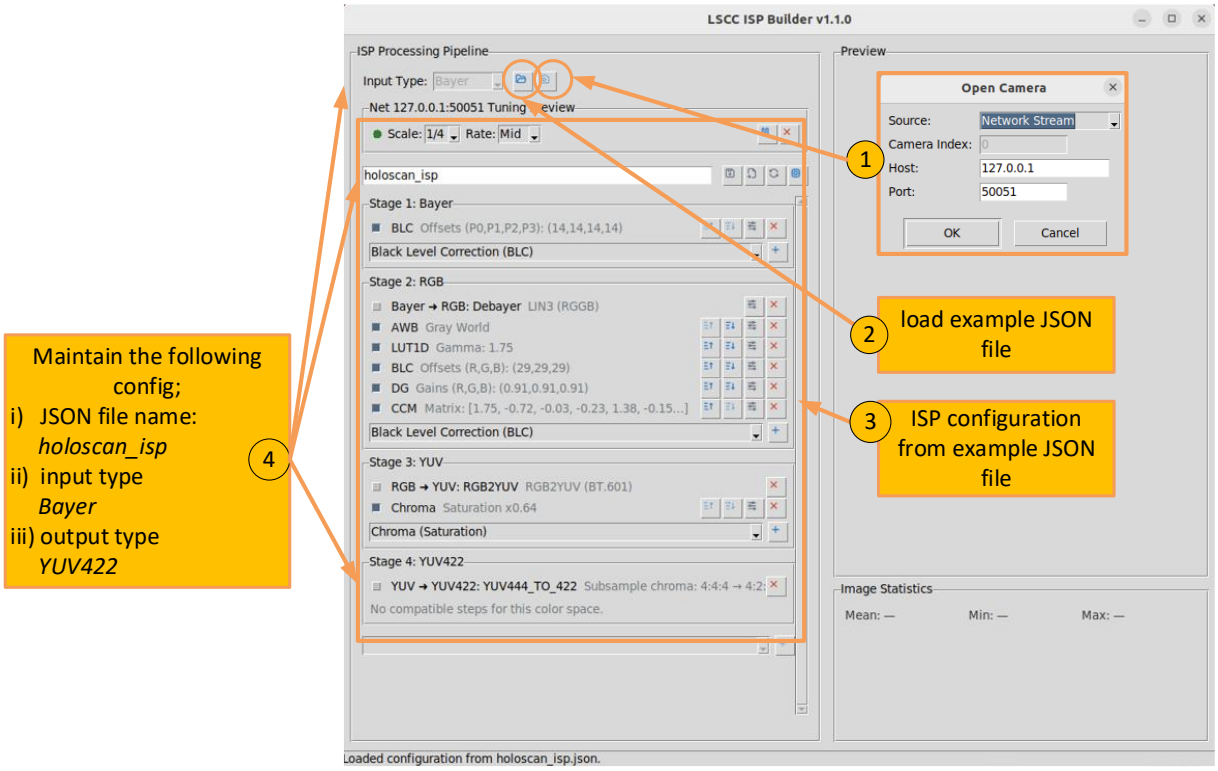


Figure 4.4.LSCC ISP Builder Linux Orin GUI.

Step 2. Enable Live Image Streaming Preview

Ensure Step 1 is complete before proceeding. Run the following commands on the Orin or Thor host terminal.

```
cd holoscan-sensor-bridge
sh docker/build.sh --igpu
xhost +
sh docker/demo.sh
python examples/linux_socketop_imx258.py --runtime-mipi=0 --focus=-200
```

A live streaming preview is displayed on the GUI as shown in Figure 4.5.

Step 3. Tune the ISP Configuration

Use the GUI to *add* or *remove* ISP processing functions and adjust the parameters of each ISP function using the available function icons (see Figure 4.5).

Configure: Click the *Configure button* ( ) beside a selected step to adjust its parameters.

Reorder: Use the *Move Up* ( ) and *Move Down* ( ) controls to modify processing order.

Add: Select a step from the drop-down menu at the bottom of each color spare and click *Add* ( ).

Remove: Click the *Remove button* ( ) to delete an unneeded step.

The live streaming image serves as the benchmark for evaluating and optimizing ISP processing functionality as enhancements are applied and tuned.

Notes:

1. A very low frame rate is used during ISP image streaming tuning. Once the desired ISP configuration is finalized, save the ISP settings to a JSON file for IP generation.
2. The changes to the video streaming preview are executed by the ISP software model. The ISP configuration setting is tuned using the ISP software model; then the tuned configuration is used for ISP IP generation. There might be a slight difference in image quality between the tuned image preview done by ISP software model compared to the actual ISP IP hardware implementation due to fixed-point precision and rounding behavior in hardware.

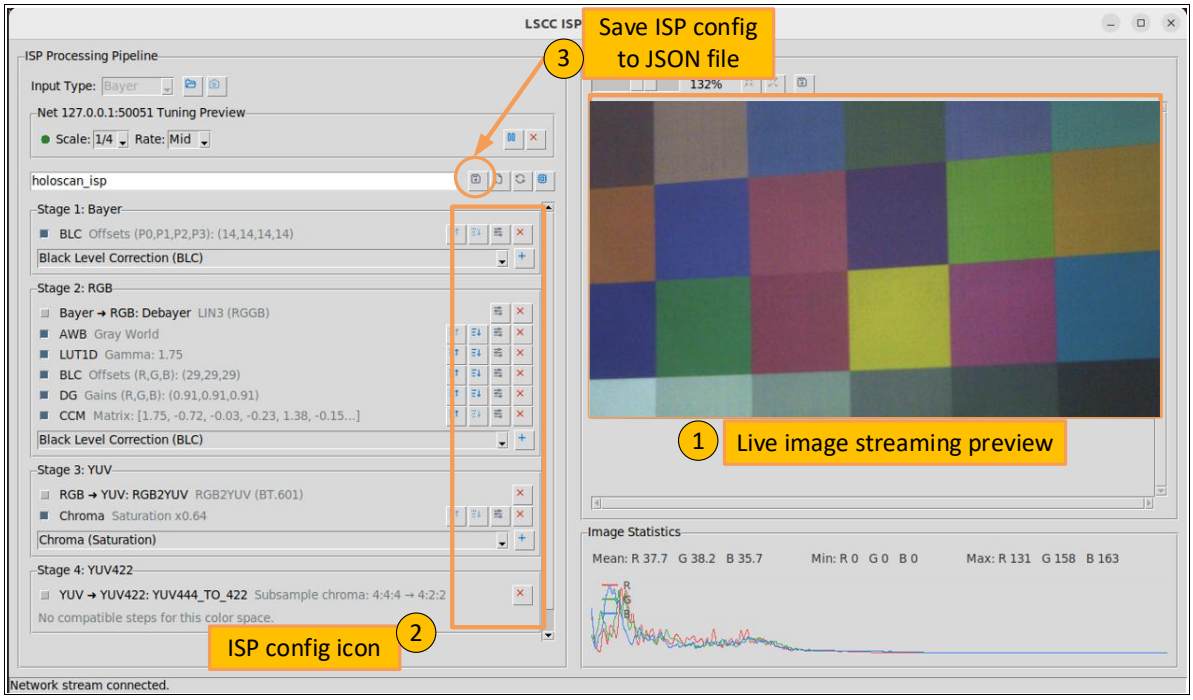


Figure 4.5. ISP builder GUI with Live Image Streaming

4.1.2.3.3. ISP IP Generation

Note: The LSCC ISP Builder (lscc_isp_builder.exe) tool must run on a local PC with Windows OS and the Lattice Radiant software installed. The tool is included as part of the reference design and is located in `/radiant/src/ip/ISP/mISP/lscc_isp_builder/` directory.

To generate the ISP IP:

1. Launch lscc_isp_builder.exe on the local PC.
2. When the GUI opens, load the configured JSON file (holoscan_isp.json). The configured ISP settings are displayed in the GUI, as shown in Figure 4.6.
3. Click **Generate Configuration**. A pop-up window appears as shown in Figure 4.7.
4. Specify the Lattice Radiant software installation path and the output directory for the generated ISP IP. The recommended output path is
   `radiant\src\ip\ISP\mISP\lscc_isp_builder\generated\holoscan_isp\isp_gen\`
5. Click **Generate RTL**.

Note: Choosing a path other than the recommended directory requires changes to the Lattice Radiant project file.

The generated ISP IP is produced in encrypted format. Output log files, and the JSON file used for generation are placed in the generated folder as shown in Figure 4.3.

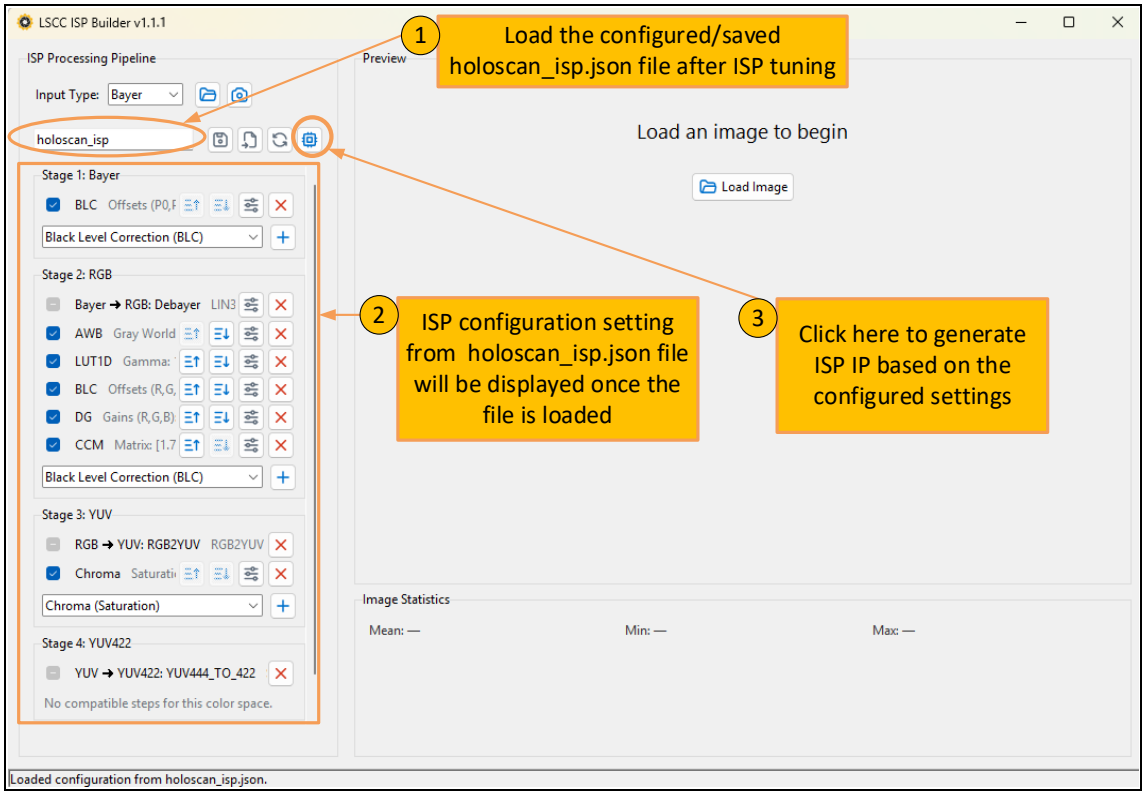


Figure 4.6. LSCC ISP Builder GUI preview loading the configured JSON file.

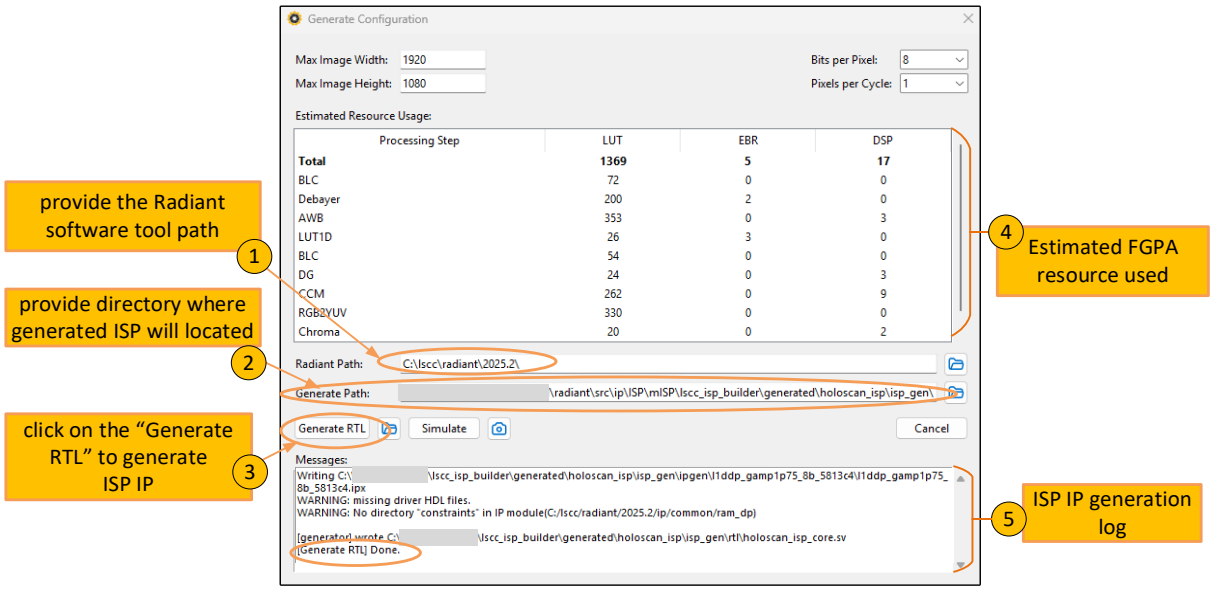


Figure 4.7. Generate Configuration window

4.1.2.4. MJPEG Reference Design IP

The MJPEG IP is a Lattice Semiconductor reference design IP provided in encrypted form and integrated as part of the reference design. Its purpose is to compress YUV422 video streams to reduce the overall video bandwidth for systems with low throughput requirement. The RTL of MJPEG IP is located in the radiant/src/ip/ISP/MJPEG/ directory.

Specifications and constraints:

- Input format: YUV422 only
- Clock domain: Single pixel clock, fixed at 160 MHz
- Default resolution: 1920 × 1080 (Full HD)
- Compression: Baseline JPEG using standard quantization and Huffman tables per JPEG ITU-T Recommendation T.81.
- Compression ratio: Approximately 40:1 for natural scenes

Note: The underlying theory of MJPEG operation is outside the scope of this document.

Due to system clock limitations on the targeted FPGA device, the pixel clock is fixed at 160 MHz. When using a four-lane MIPI sensor, the MIPI lane rate must be 400 Mbps or less to support Full HD.

Note: The MJPEG format is currently limited to an output frame rate of 30 fps in this release.

4.1.2.5. Hololink IP

The Hololink IP is developed and maintained by NVIDIA. The latest Hololink IP source code is available of GitHub at the following link, GitHub - nvidia-holoscan/holoscan-sensor-bridge at 2.5.0. Currently, the reference design integrates the latest available Hololink IP, which is version 2511.

4.1.2.6. Ethernet IP Generation

Specific Ethernet IPs are required to support 10GbE or 25GbE link speeds in the Avant-X device. To support a 10GbE link speed, the 10Gb Ethernet (MAC + PHY) IP Core v3.3.1 is used. To support Ethernet 25GbE link speed, the 25Gb Ethernet (MAC + PHY) IP Core v2.2.1 is used. The 10GbE and 25GbE Ethernet Lane IDs is set to *auto* for the Avant-X Versa Board before generating the IP. Figure 4.8. shows the 10GbE Ethernet IP generation, while Figure 4.9 shows the Ethernet 25GbE MAC+PHY IP generation.

Note: IP version may vary as they are updated periodically. The figures below are for illustration purposes.

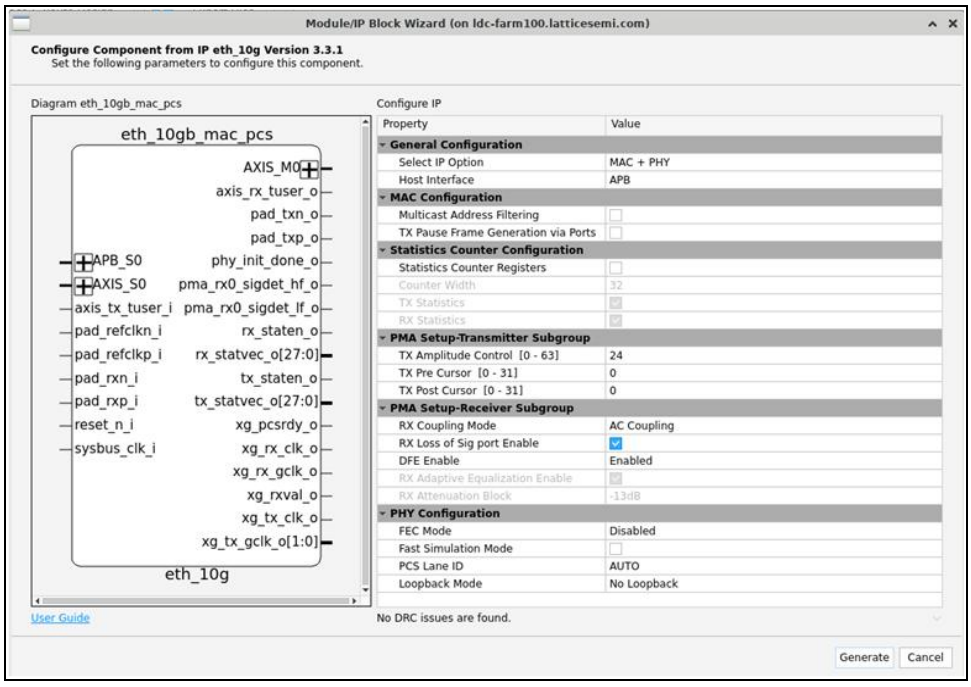


Figure 4.8. Ethernet 10GbE (MAC + PHY) IP Generation

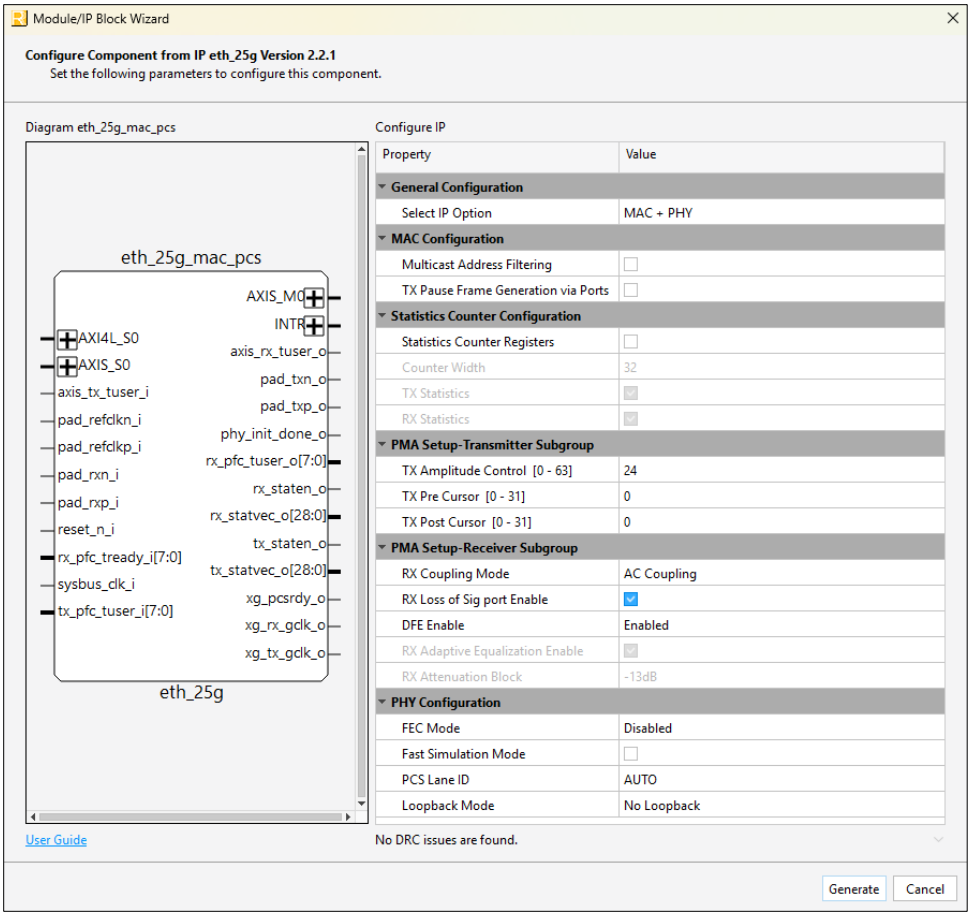


Figure 4.9. Ethernet 25GbE (MAC + PHY) IP Generation

4.1.3. Configuring the HDL Parameter and DEFINE Switch

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
- MIPI CSI-2 RX D-PHY configuration
  - Method: Preset MIPI CSI-2 RX D-PHY IP configuration (configure the number of lanes per camera sensor and the MIPI lane rate used for each camera sensor)
- Image preprocessing
  - ISP decoding or MJPEG compression on (enabled by default)
- Number of cameras 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: 10GbE or 25GbE
- Number of Ethernet links used as host interface
- Ethernet PCS lane ID configuration

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

```
<project_name>/fpga_lavat/radiant/src/rtl/hsb_avtx/top/versa_top/HOLOLINK_def.svh
```

Configurations such as camera 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 10GbE or 25GbE links. Figure 4.10 shows the DEFINE switches that need to be changed in the 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.

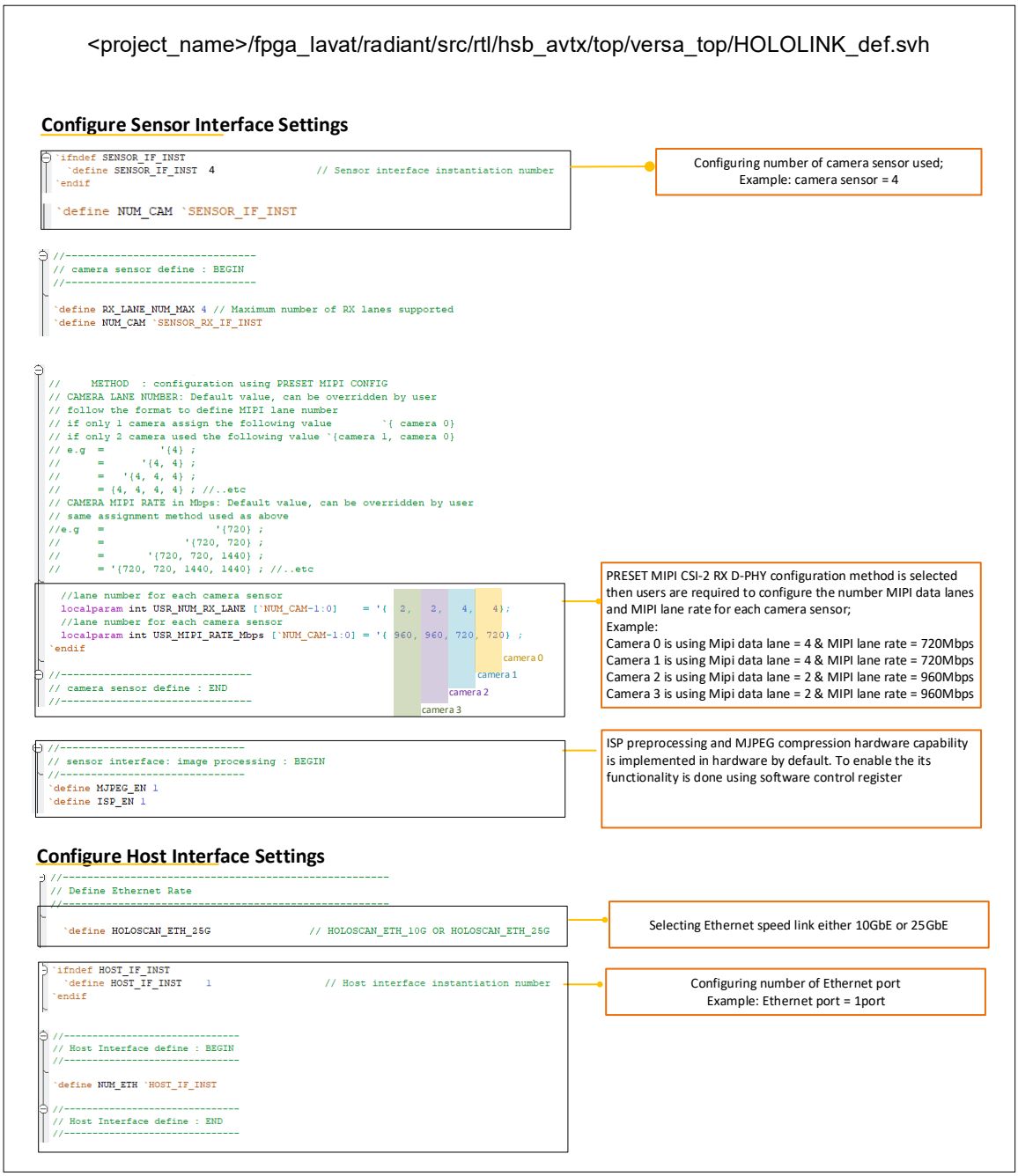


Figure 4.10. Define Switch Setting in HOLOLINK_def.svh File

Note: Figure 4.10 shows an example of setting 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] will change. Therefore, the data assigned to it are trimmed or added based on the intended camera number usage.

4.1.4. Configuring the Platform Design Constraint

Once the HDL design configuration is finalized, platform design constraints in both the SDC and PDC files are updated. The reference design include SDC and PDC template files; their locations are listed in Table 4.3.

Table 4.3. PDC Template Files

Reference design	PDC template file location
Avant-X Versa Board	<project_name>/fpga_lavat/radiant/src/constraints/hsb_avtx_versa.sdc
	<project_name>/fpga_lavat/radiant/src/constraints/hsb_avtx_versa.pdc

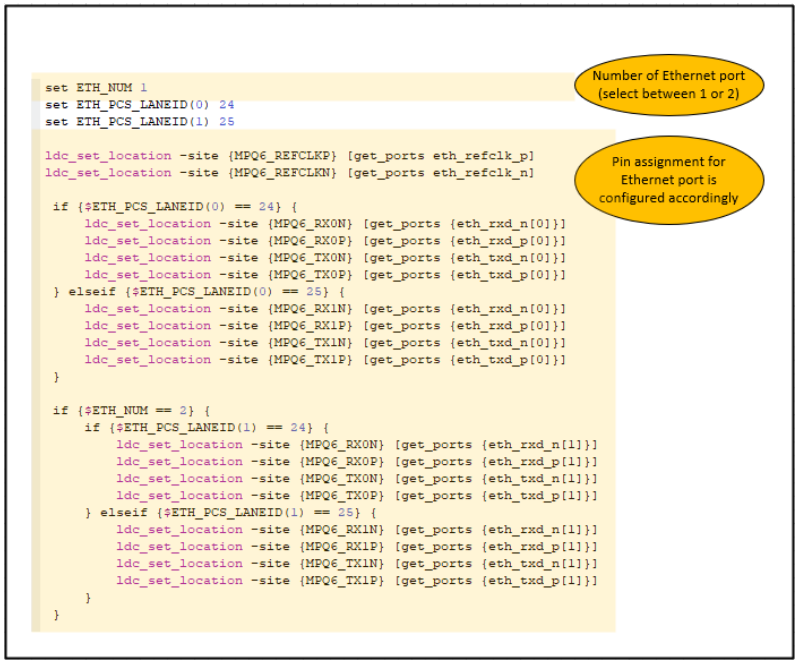
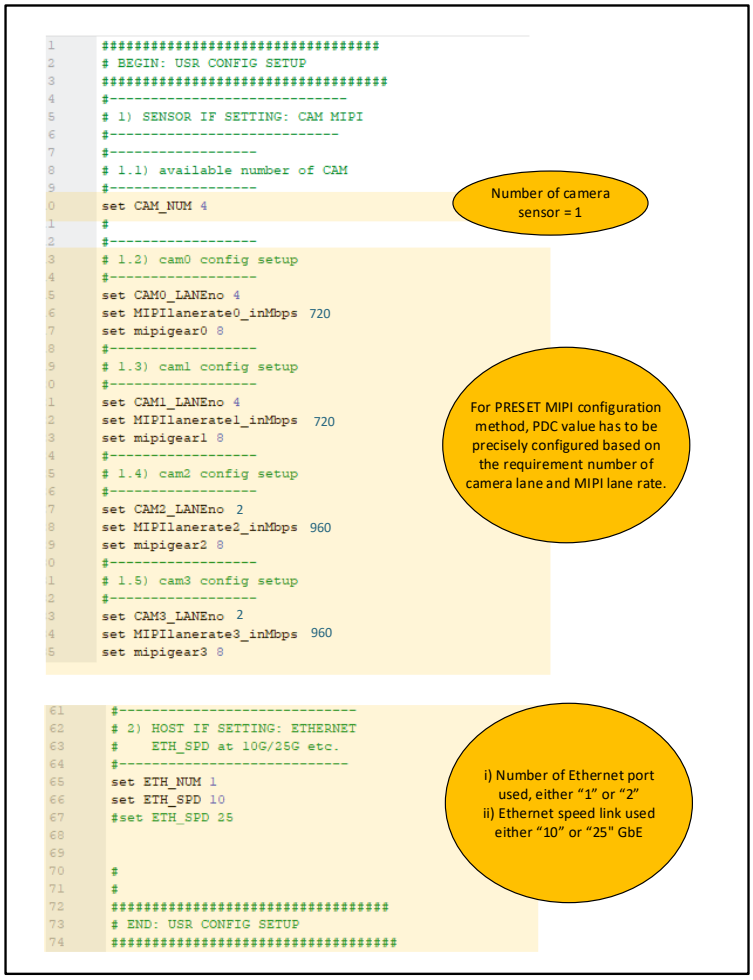


Figure 4.11. SDC File Setting for Ethernet Ports and Pin Allocation

Figure 4.11 shows the SDC file settings for the Ethernet port configuration and pin allocation. The Avant-X Versa Board supports a maximum of two SFP+ connectors.




```
#################################
# BEGIN: USR CONFIG SETUP
#################################
#-----------------------------
# 1) SENSOR IF SETTING: CAM MIPI
#-----------------------------
#------------------
# 1.1) available number of CAM
#------------------
set CAM_NUM 4
#
#------------------
# 1.2) cam0 config setup
#------------------
set CAM0_LANEno 4
set MIPIlanerate0_inMbps 720
set mipigear0 8
#------------------
# 1.3) cam1 config setup
#------------------
set CAM1_LANEno 4
set MIPIlanerate1_inMbps 720
set mipigear1 8
#------------------
# 1.4) cam2 config setup
#------------------
set CAM2_LANEno 2
set MIPIlanerate2_inMbps 960
set mipigear2 8
#------------------
# 1.5) cam3 config setup
#------------------
set CAM3_LANEno 2
set MIPIlanerate3_inMbps 960
set mipigear3 8

#-----------------------------
# 2) HOST IF SETTING: ETHERNET
#    ETH_SPD at 10G/25G etc.
#-----------------------------
set ETH_NUM 1
set ETH_SPD 10
#set ETH_SPD 25


#
#
#################################
# END: USR CONFIG SETUP
#################################
```

Figure 4.12. PDC File Setting for Camera and Ethernet Configuration

Figure 4.12 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 10GbE or 25GbE. If fewer cameras are used, the lane numbers for undefined cameras are ignored. For example, if the number of cameras is 1, the lane number values set for cameras 2, 3, and 4 are 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 the intended reference design. However, the PDC file template provided with the reference design allows you to choose 1, 2, 3, or 4 camera sensors.

4.1.4.1. 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 Avant-X 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 top level uses a two-dimensional array coding method. This affects how the assignments are defined and mapped in the PDC file. Table 4.4 illustrates the correct pin assignment approach under these conditions in the PDC file up to four cameras.

Table 4.4. PDC Pin Assignment for 2D 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_n[0][2]	mipi_cam_data_n[2]
	mipi_cam_data_p[0][3]	mipi_cam_data_p[3]
	mipi_cam_data_n[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_n[1][2]	mipi_cam_data_n[6]
	mipi_cam_data_p[1][3]	mipi_cam_data_p[7]
	mipi_cam_data_n[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_n[2][2]	mipi_cam_data_n[10]
	mipi_cam_data_p[2][3]	mipi_cam_data_p[11]
	mipi_cam_data_n[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_n[3][2]	mipi_cam_data_n[14]
	mipi_cam_data_p[3][3]	mipi_cam_data_p[15]
	mipi_cam_data_n[3][3]	mipi_cam_data_n[15]

4.1.4.2. Timing constraint for MIPI CSI-2 Camera Sensor Interface

Using the PDC template from the 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 4.12. In the PDC file, this value will be used to assign the timing constraint required for the MIPI clock input.

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 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 4.13 shows a screenshot of the reference design implemented on the Avant-X Versa Board. If the verified design is not the target reference design, review the HDL parameters, DEFINE switch configuration, and PDC configuration.

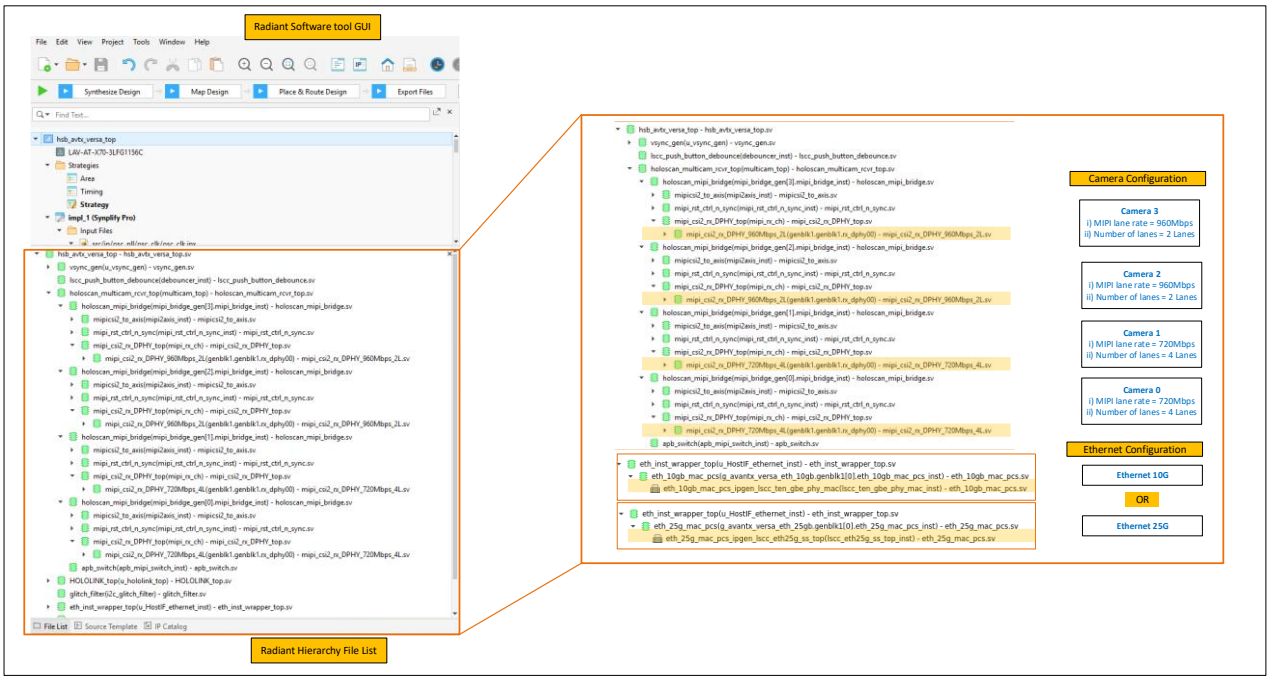


Figure 4.13. Screenshot of Lattice Radiant Software Hierarchy File List

4.1.6. Enabling ISP or MJPEG Feature at Runtime

By default, ISP processing and MJPEG compression are supported in hardware. You can confirm this by checking that the parameters ISP_EN and MJPEG_EN are set to 1 in the *HOLOLINK_def.svh* file. To conserve FPGA resources, you can disable these parameters (set to 0) to meet your application resource requirements.

The ISP_EN parameter controls whether the ISP hardware instance is enabled, while MJPEG_EN controls the MJPEG compression hardware. Table 4.5 shows the relationship between these parameters in hardware enablement.

Table 4.5. ISP or MJPEG Parameters Enablement in Hardware

Configuration Mode	ISP_EN	MJPEG_EN	Description
ISP only	1	0	Enables ISP processing only. MJPEG compression is disabled.
ISP + MJPEG	1	1	Enables both ISP processing and MJPEG compression. MJPEG depends on ISP and requires it to be enabled.
ISP disabled	0	X (disabled)	Disables ISP. MJPEG is automatically unavailable and its setting is ignored.

When the ISP processing is enabled in hardware (ISP_EN = 1), with or without MJPEG (MJPEG_EN = 0 or 1), the software controls ISP and MJPEG operation through control registers. Provide the appropriate argument when running the player command-line tool. For further details on command-line options, refer to the Testing the System section.

4.1.7. Configuring the Driver and Applying the Patch

The driver configuration must be adapted for the specific reference design. Because each camera sensor uses distinct I2C control signals, the driver must be updated to correctly detect and communicate with the intended sensor, ensuring proper initialization and operation.

Configuration support for the IMX258 sensor is described in the Building the Holoscan Sensor Bridge Demo Container section. To enable this support, follow the procedure outlined from Configuring the AGX Orin Developer Kit through the Building the Holoscan Sensor Bridge Demo Container section.

4.2. Customizing the Reference Design for the Avant-X Versa Board

Figure 4.14 shows an overview of a configured Avant-X Versa board block diagram.

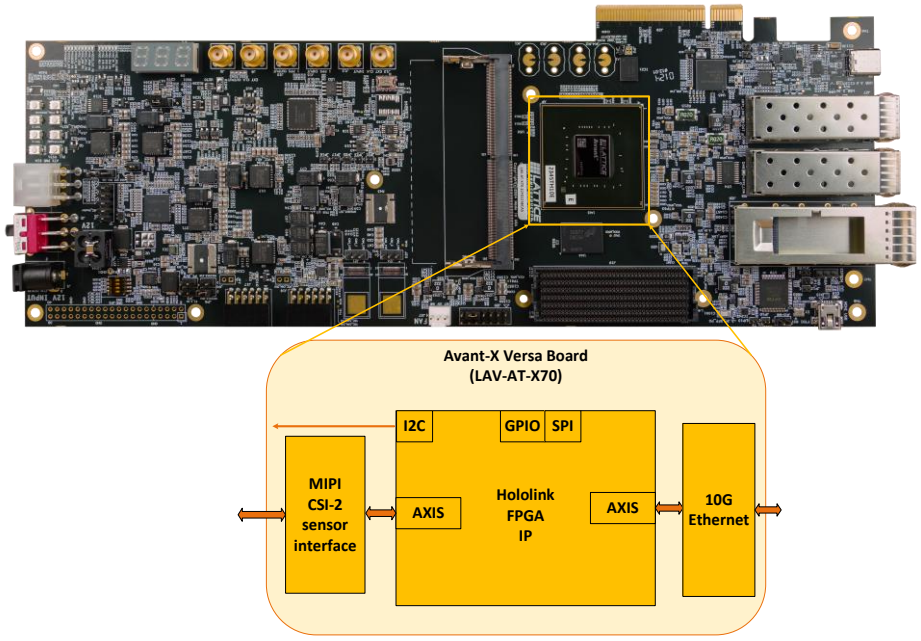


Figure 4.14. Avant-X Versa Board Block Diagram

Table 4.6 shows the configuration required for the reference design to fit on the Avant-X Versa Board.

Table 4.6. Configuration for the Reference Design to fit into the Avant-X Versa Board

Sensor/Host Interfaces	Features		Configuration
Camera sensor interface	Number of camera sensors		1
	Preset MIPI configuration method		—
	1	Camera 0: MIPI lane number	4
	2	Camera 0: MIPI lane rate	720 Mbps
Ethernet Link	Number of Ethernet port		1
	Ethernet link speed		10GbE

To customize the reference design, follow the steps below.

Note: This reference design is already integrated with a preset MIPI RX D-PHY IP configuration, the latest Hololink IP version 2511, and Ethernet IPs.

1. Customize the reference design to match the Avant-X Versa Board specifications.
2. Configure HDL Parameter and DEFINE Switch.

   The reference design needs to be configured by changing the settings of the HDL parameter and DEFINE switch to meet the Avant-X Versa Board requirement using this file.

   `<project_name>/fpga_lavat/radiant/src/rtl/hsb_avtx/top/versa_top/HOLOLINK_def.svh`

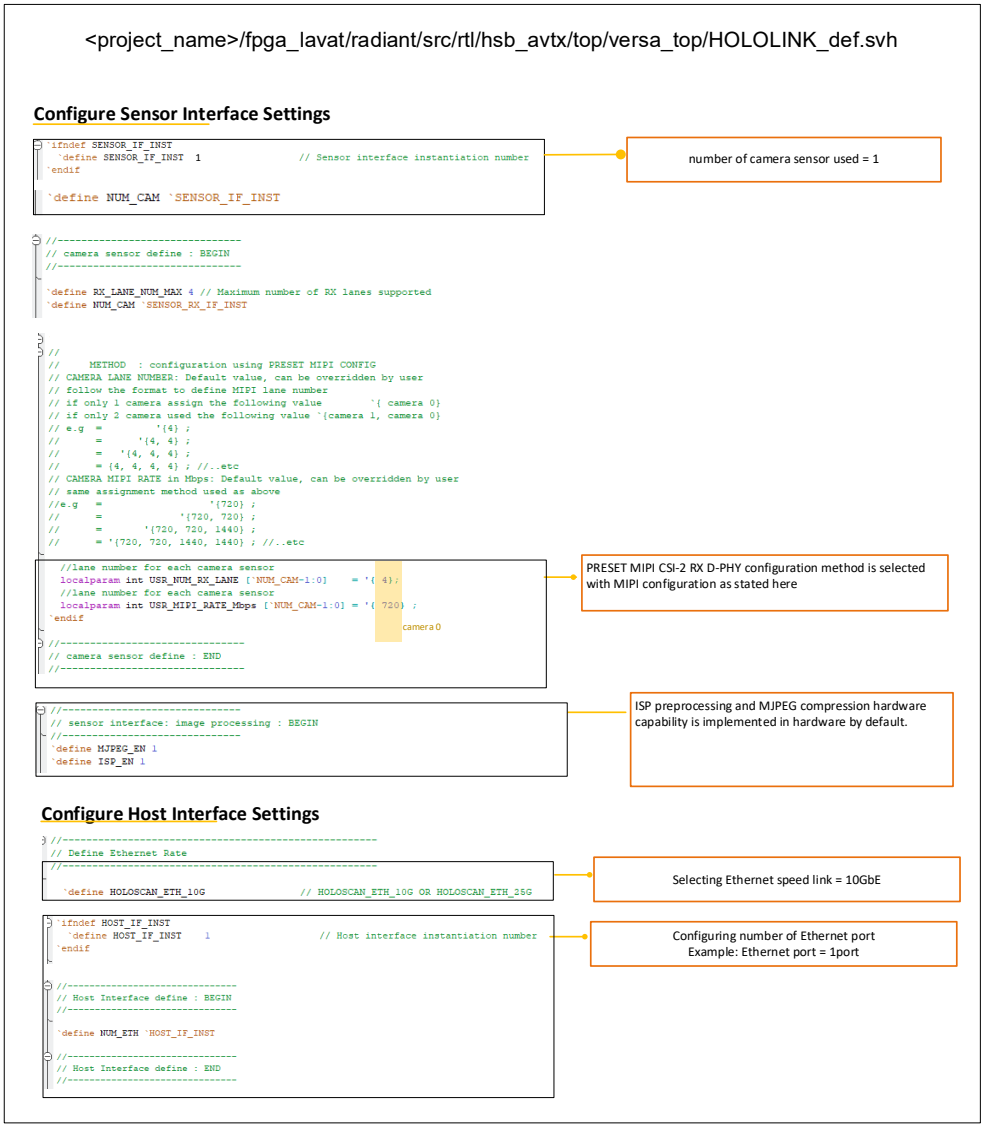
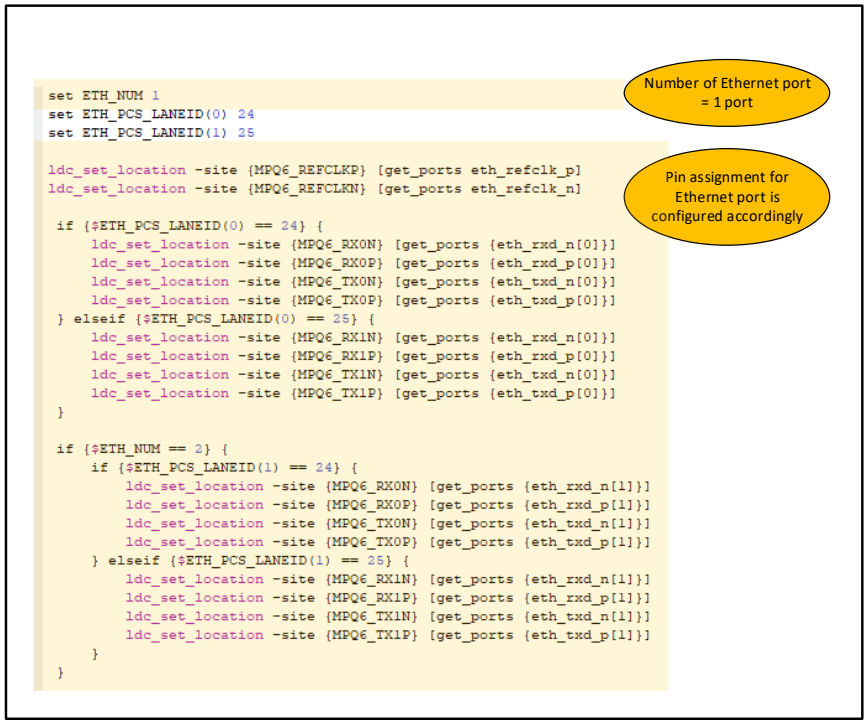


Figure 4.15. Required HDL Parameter and DEFINE Switch Configuration

3. Review the PDC configuration file.

   Figure 4.16 shows the screenshot of the required SDC configuration and Figure 4.17 shows a screenshot of the required PDC configuration.

   `<project_name>/fpga_lavat/radiant/src/constraints/hsb_avtx_versa.sdc`
   `<project_name>/fpga_lavat/radiant/src/constraints/hsb_avtx_versa.pdc`




```
set ETH_NUM 1
set ETH_PCS_LANEID(0) 24
set ETH_PCS_LANEID(1) 25

ldc_set_location -site {MPQ6_REFCLKP} [get_ports eth_refclk_p]
ldc_set_location -site {MPQ6_REFCLKN} [get_ports eth_refclk_n]

 if {$ETH_PCS_LANEID(0) == 24} {
     ldc_set_location -site {MPQ6_RX0N} [get_ports {eth_rxd_n[0]}]
     ldc_set_location -site {MPQ6_RX0P} [get_ports {eth_rxd_p[0]}]
     ldc_set_location -site {MPQ6_TX0N} [get_ports {eth_txd_n[0]}]
     ldc_set_location -site {MPQ6_TX0P} [get_ports {eth_txd_p[0]}]
 } elseif {$ETH_PCS_LANEID(0) == 25} {
     ldc_set_location -site {MPQ6_RX1N} [get_ports {eth_rxd_n[0]}]
     ldc_set_location -site {MPQ6_RX1P} [get_ports {eth_rxd_p[0]}]
     ldc_set_location -site {MPQ6_TX1N} [get_ports {eth_txd_n[0]}]
     ldc_set_location -site {MPQ6_TX1P} [get_ports {eth_txd_p[0]}]
 }

 if {$ETH_NUM == 2} {
     if {$ETH_PCS_LANEID(1) == 24} {
         ldc_set_location -site {MPQ6_RX0N} [get_ports {eth_rxd_n[1]}]
         ldc_set_location -site {MPQ6_RX0P} [get_ports {eth_rxd_p[1]}]
         ldc_set_location -site {MPQ6_TX0N} [get_ports {eth_txd_n[1]}]
         ldc_set_location -site {MPQ6_TX0P} [get_ports {eth_txd_p[1]}]
     } elseif {$ETH_PCS_LANEID(1) == 25} {
         ldc_set_location -site {MPQ6_RX1N} [get_ports {eth_rxd_n[1]}]
         ldc_set_location -site {MPQ6_RX1P} [get_ports {eth_rxd_p[1]}]
         ldc_set_location -site {MPQ6_TX1N} [get_ports {eth_txd_n[1]}]
         ldc_set_location -site {MPQ6_TX1P} [get_ports {eth_txd_p[1]}]
     }
 }
```

Figure 4.16. Required SDC Configuration


```
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_LANEno 4
16  set MIPIlanerate0_inMbps 720
17  set mipigear0 8
18  #------------------


61  #-----------------------------
62  # 2) HOST IF SETTING: ETHERNET
63  #    ETH_SPD at 10G/25G etc.
64  #-----------------------------
65  set ETH_NUM 1
66  set ETH_SPD 10
67  #set ETH_SPD 25
68
69
70  #
71  #
72  ###################################
73  # END: USR CONFIG SETUP
74  ###################################
```

Figure 4.17. Required PDC Configuration

4. Configure the driver: For details on how to configure the driver, refer to the Building the Holoscan Sensor Bridge Demo Container section. To enable driver configuration support, follow the steps from the Configuring the AGX Orin Developer Host Kit section through the Building the Holoscan Sensor Bridge Demo Container section.
5. 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 4.18 shows the reference design in the Lattice Radiant software hierarchy file list. The reference design is now ready for compilation and implementation.

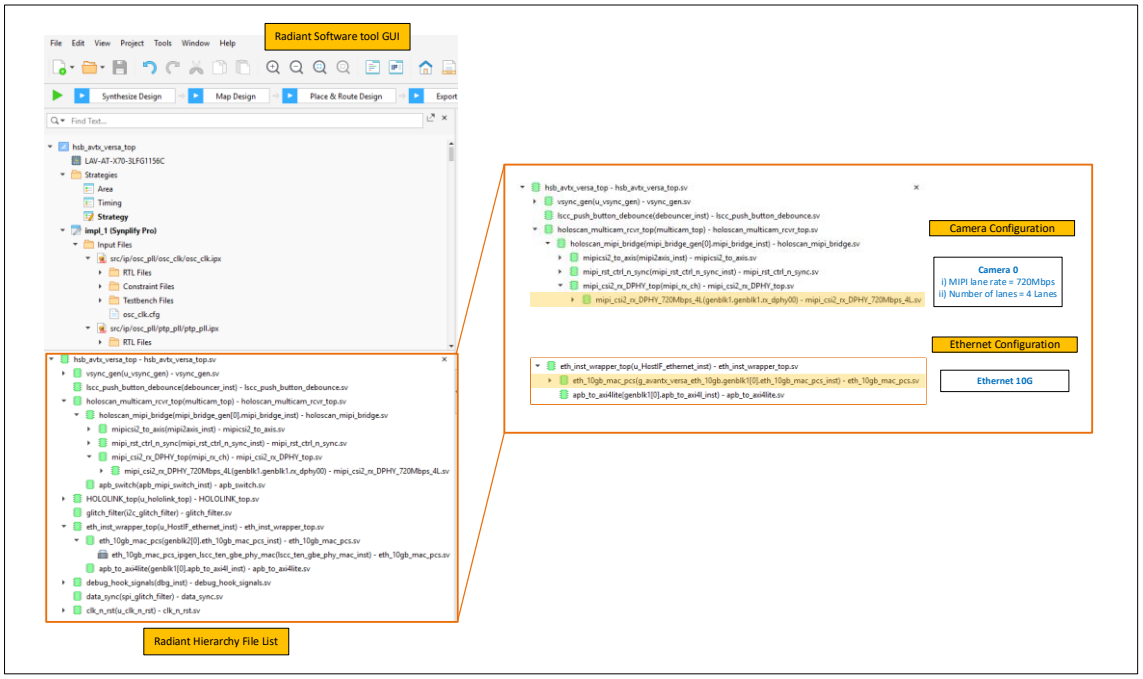


Figure 4.18. Lattice Radiant Software Hierarchy File

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, refer to the Lattice Radiant User Guide. The screenshots provided in this section are for reference only and the details may vary, such as the version of the software, FW ID, Board ID, and date code being used.

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 for Each Bitstream

To assign a revision ID for each bitstream, modify the reference design HDL top-level file, *hsb_avtx_versa_top.sv* as shown in Figure 5.1.

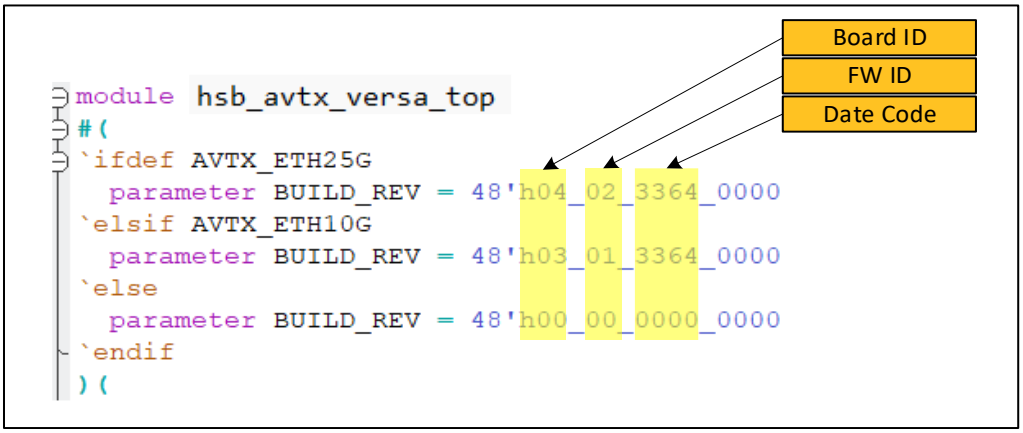


Figure 5.1. Revision ID assignment in hsb_avtx_versa_top.sv

The Board ID is stored in a 1-byte hexadecimal value determined by the HSB platform and the configured Ethernet link speed. Table 5.1 lists the Board ID values for each supported HSB platform and link-speed combination. The firmware (FW) ID is also stored in a 1-byte hexadecimal value and increments with each FPGA bitstream release.

Table 5.1. Board ID Representation in Hexadecimal

HSD Board with Ethernet Link Speed	Board ID
Avant-X Versa board with Ethernet speed 10 Gbps	0x03
Avant-X Versa board with Ethernet speed 25 Gbps	0x04

The date code indicates when the release bitstream was generated. It is stored in a 2-byte hexadecimal format, as shown in Figure 5.2. The figure provides an example that converts the date November 4, 2025 from decimal to binary and then to hexadecimal formats.

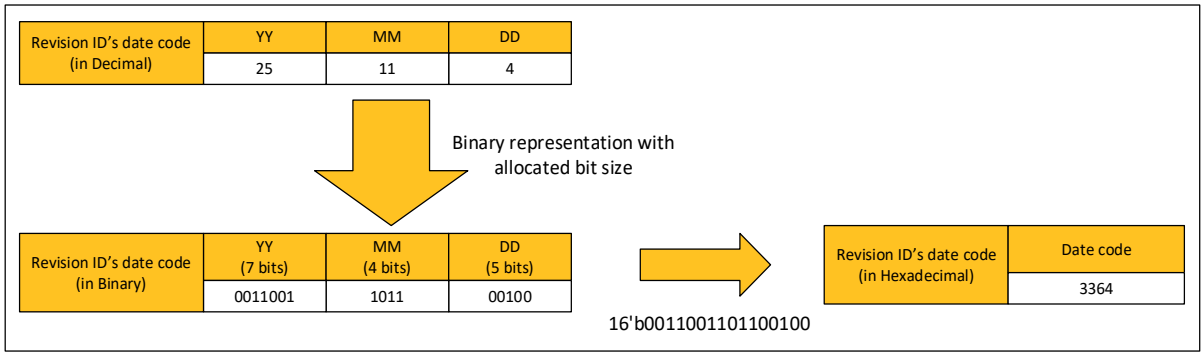


Figure 5.2. Revision ID Date Code Representation

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 Lattice Radiant software.

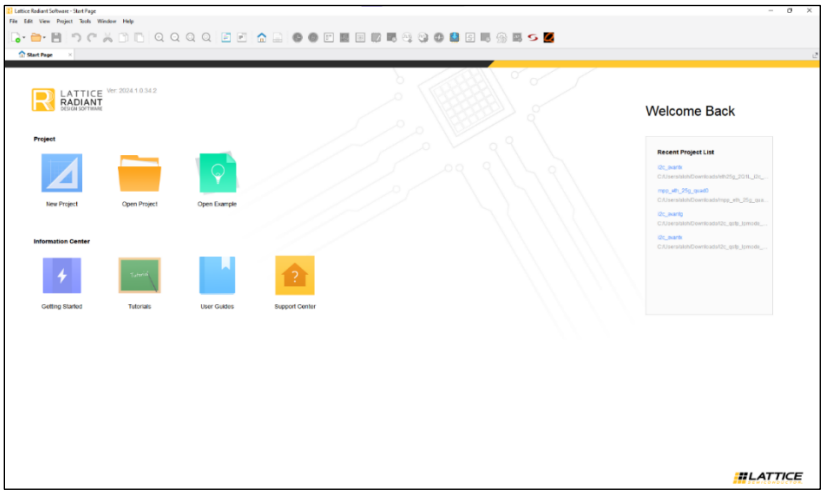


Figure 5.3. Lattice Radiant Software

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

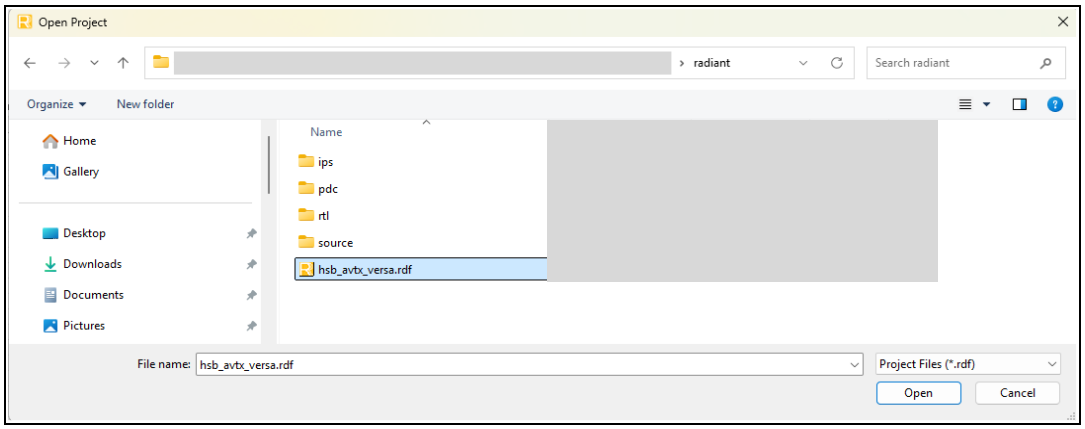


Figure 5.4. Open Project File

3. Modify the Synplify Pro® strategy details under **Synthesis Design** > **Synplify Pro**. Refer to Figure 5.5.

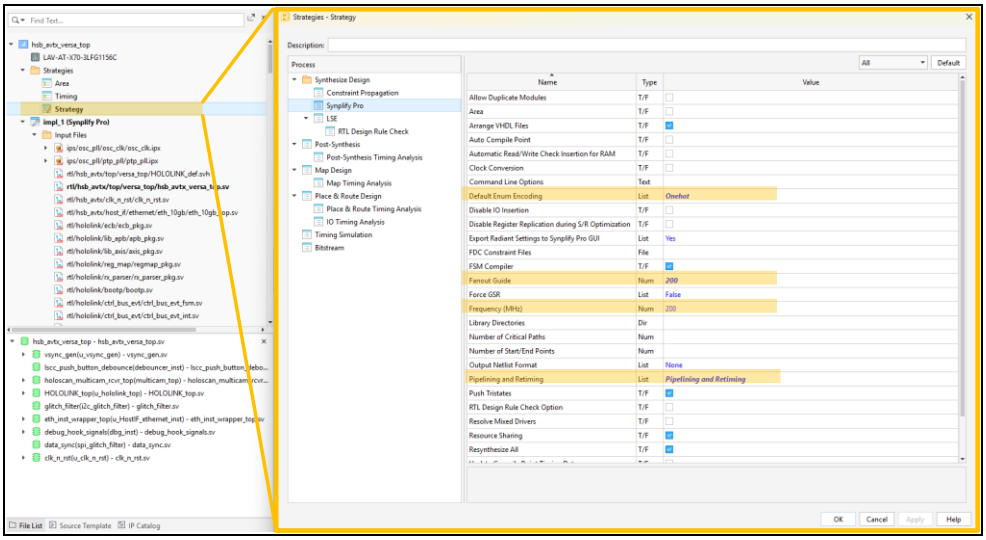


Figure 5.5. Synthesis Design Strategy

4. Modify the Place and Route design strategy details under **Strategy1** > **Place & Route Design**. Refer to Figure 5.6. The seed run count is set to 10 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 implementation[1].

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.

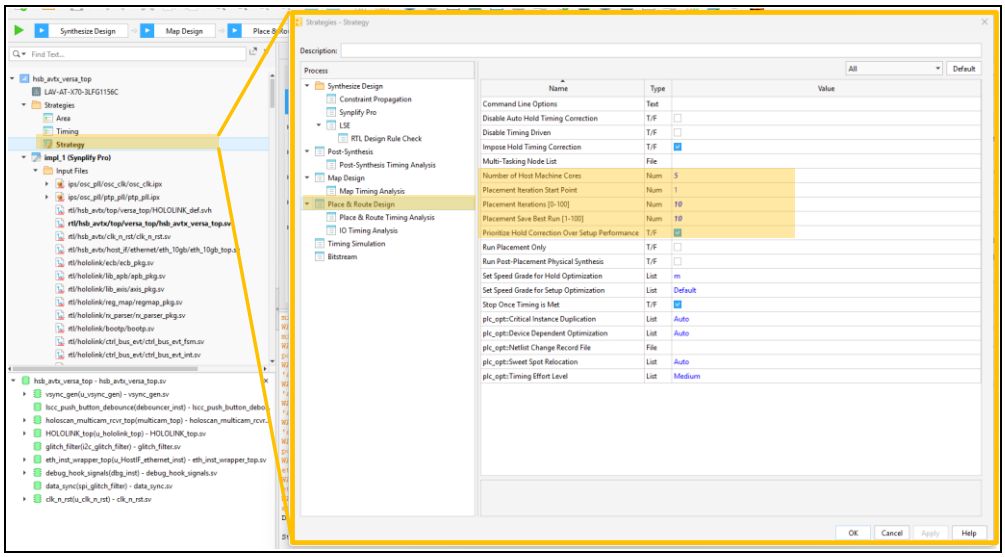


Figure 5.6. Place and Route Design Strategy

For further details on configuring the Lattice Radiant Software, refer to the Lattice Radiant User Guide.

5.4. Generating the Bitstream

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.

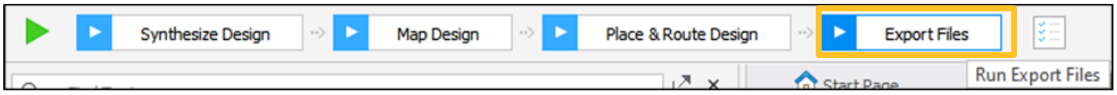


Figure 5.7. Generate and Export Bitstream File

6. Implementing the Reference Design

The following table provides an overview of the compatibility of Avant-X Versa Board with various NVIDIA Jetson modules and their respective Ethernet speed capabilities.

Table 6.1. Avant-X Platform and Jetson Module Compatibility

Board Type	Jetson Module	
	AGX Orin	AGX Thor
Avant-X Versa Board	10GbE	10GbE and 25GbE

6.1. Hardware Requirements

6.1.1. Avant-X Versa Board and NVIDIA Jetson AGX Orin

- NVIDIA Jetson AGX Orin Developer Kit
- Lattice Avant-X Versa Board
- IMX258 Camera Sensor Module – contact Lattice Sales for more information
- Ethernet 10GBase-T (Cat 6 RJ-45) SFP+ Module (Ethernet 10G support) – tested with 6COM 10GBASE SFP+ RJ-45 Module
- CAT6 Ethernet cable (support 10GbE)
- Type-C USB Cable
- DisplayPort Cable
- Monitor with DisplayPort support (for AGX Orin)
- Keyboard (for AGX Orin)
- Mouse (for AGX Orin)
- Mini USB Type-A cable for programming
- 12 V Power Supply
- Host Orin AGX power adapter

Set the additional camera power jumper connector (J22) to 1.2 V (see Figure 6.1). Use this setting for the IMX258 camera, which supports 1.0 V and 1.2 V depending on the camera module.

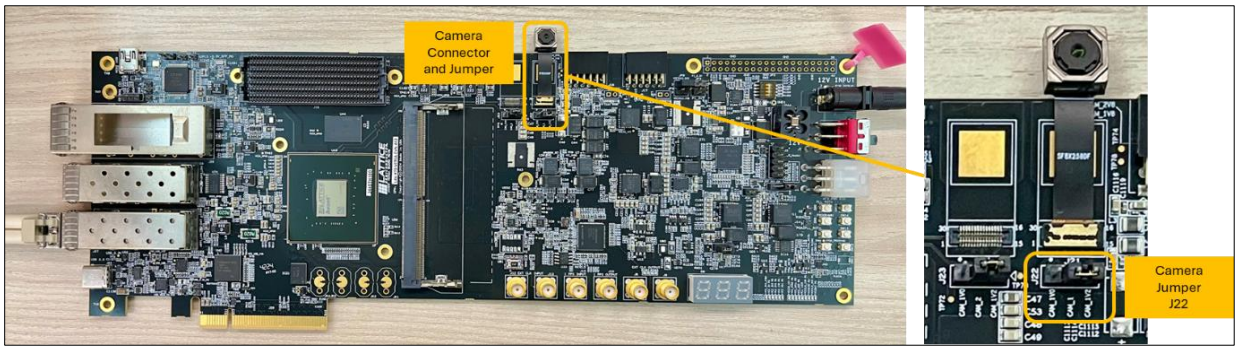


Figure 6.1. Avant-X Versa Board Camera Connector and Jumper

Figure 6.2 shows the hardware setup for the Jetson AGX Orin host and the Lattice Avant-X Versa Board used in the 10GbE Ethernet reference design implementation.

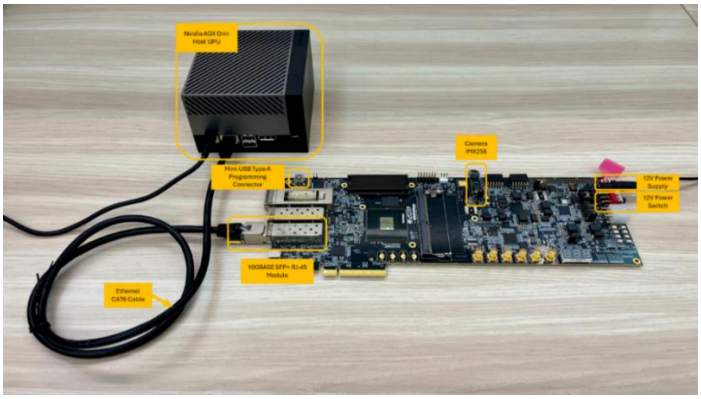


Figure 6.2. Hardware Setup to support 10GbE Ethernet

6.1.2. Avant-X Versa Board and NVIDIA Jetson AGX Thor

- NVIDIA Jetson AGX Thor Developer Kit
- Lattice Avant-X Versa Board
- 13 MP IMX258 Camera module – contact Lattice Sales for more information
- 100G QSFP28 to 4×25G SFP28 Breakout Active Optical Cable (AOC) – tested with FS 100G QSFP28 to 4×25G Breakout AOC
- Type-C USB Cable to Type-A USB Cable
- DisplayPort Cable
- Monitor with DisplayPort support (for AGX Thor)
- Keyboard (for AGX Thor)
- Mouse (for AGX Thor)
- Mini USB to USB-A Cable (for programming)
- 12 V Power Supply (for Avant-X Versa Board)
- Power adapter (for AGX Thor)

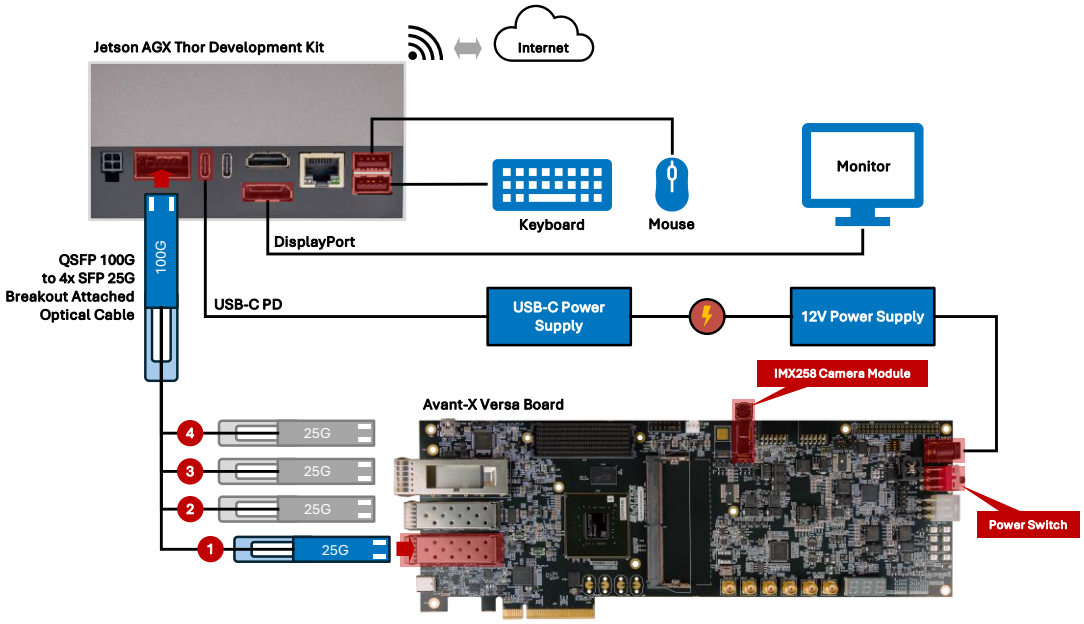


Figure 6.3. Setup with AGX Thor for 10GbE and 25GbE Ethernet

6.2. Software Requirements

6.2.1. Jetson AGX Orin

- JetPack 6.2.1
- NVIDIA Jetson Linux 36.4.4
- NVIDIA Holoscan SDK 3.7.0
- Holoscan Sensor Bridge (HSB) SDK 2.5.0

6.2.2. Jetson AGX Thor

- JetPack 7.0
- NVIDIA Jetson Linux 38.2
- NVIDIA Holoscan SDK 3.7.0
- Holoscan Sensor Bridge (HSB) SDK 2.5.0

6.3. NVIDIA Platform Host Setup

6.3.1. Jetson AGX ORIN Developer Kit

6.3.1.1. 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 the 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 6.4.

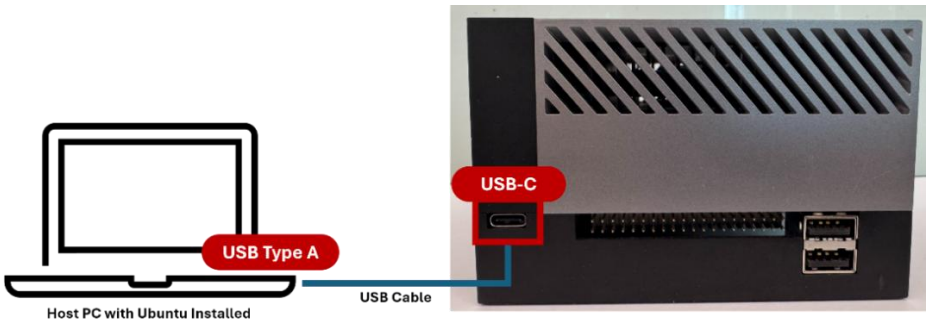


Figure 6.4. Ubuntu Host PC and Jetson AGX Orin Developer Kit Setup

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 6.5. While holding the recovery button, press the reset button and release it. Wait at least three seconds, then release the recovery button.

Figure 6.5. 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.**

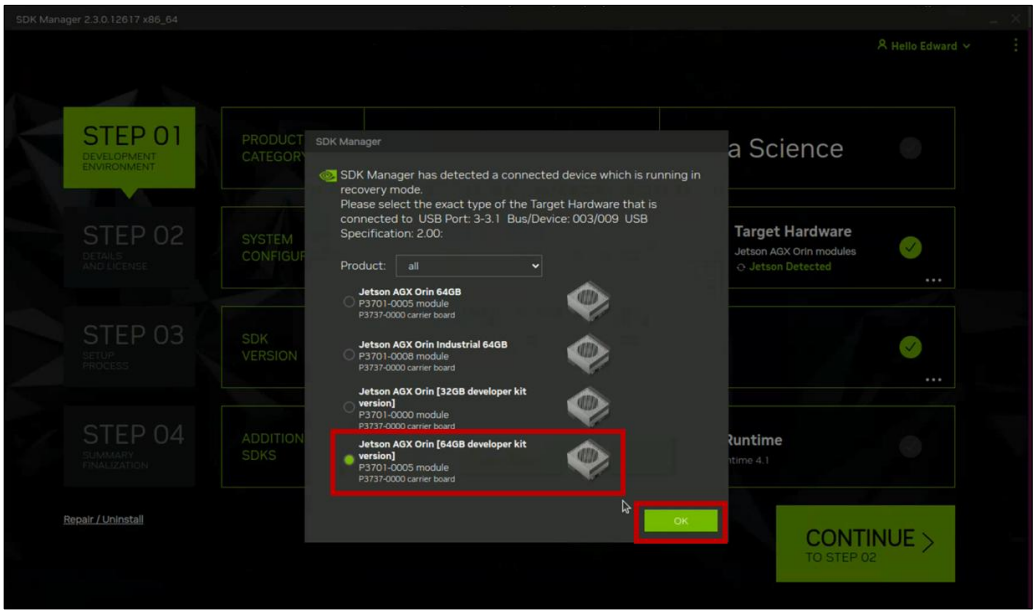


Figure 6.6. Jetson AGX Orin [64GB developer kit version]

7. Click the **…** icon at **(A)** as shown in Figure 6.7, and select **JetPack 6.2.1 (Rev. 1)**. Then, click **CONTINUE** to proceed.

Figure 6.7. 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.

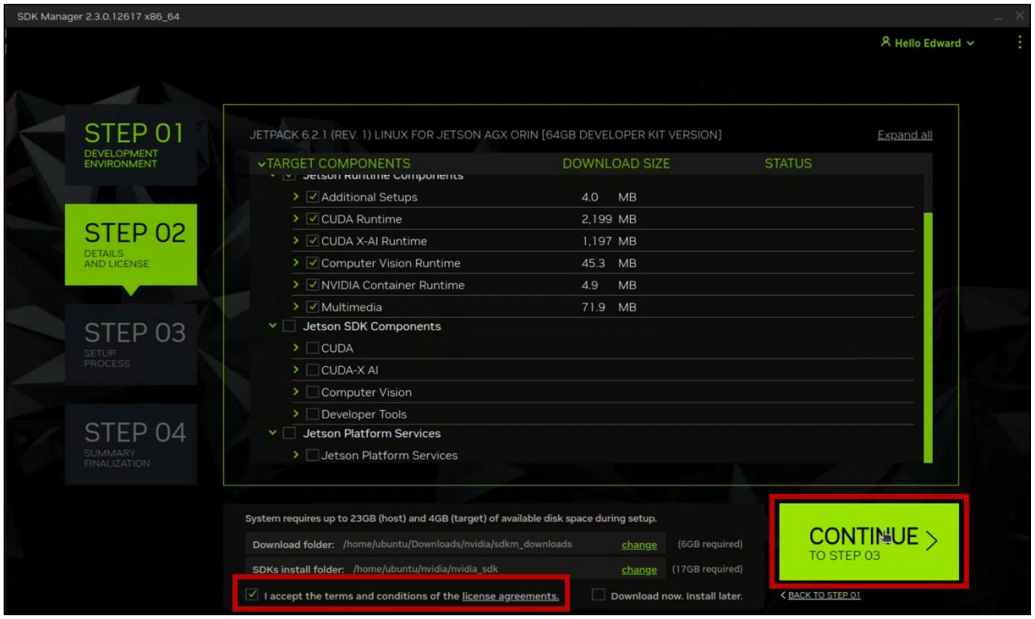


Figure 6.8. Terms and Conditions – License Agreement

9. In **STEP 03 – SETUP IN PROCESS**, as shown in Figure 6.9, wait for the download operation and OS image creation process to complete.

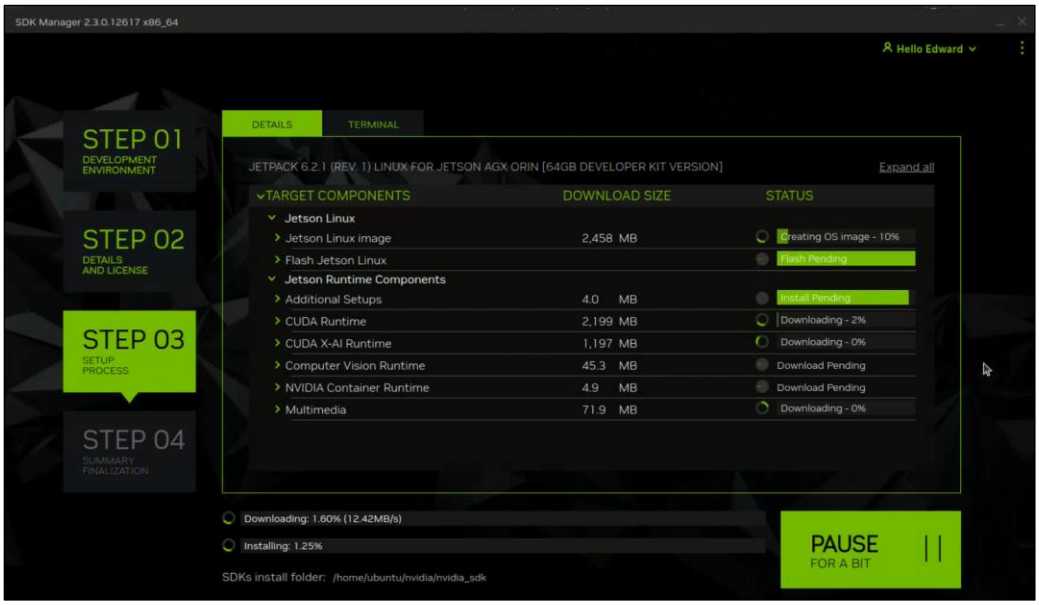


Figure 6.9. Download Operation and OS Image Creation

10. When prompted for the flash setup, enter the desired username to access the AGX Orin system, see letter **(A)** in Figure 6.10, 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.

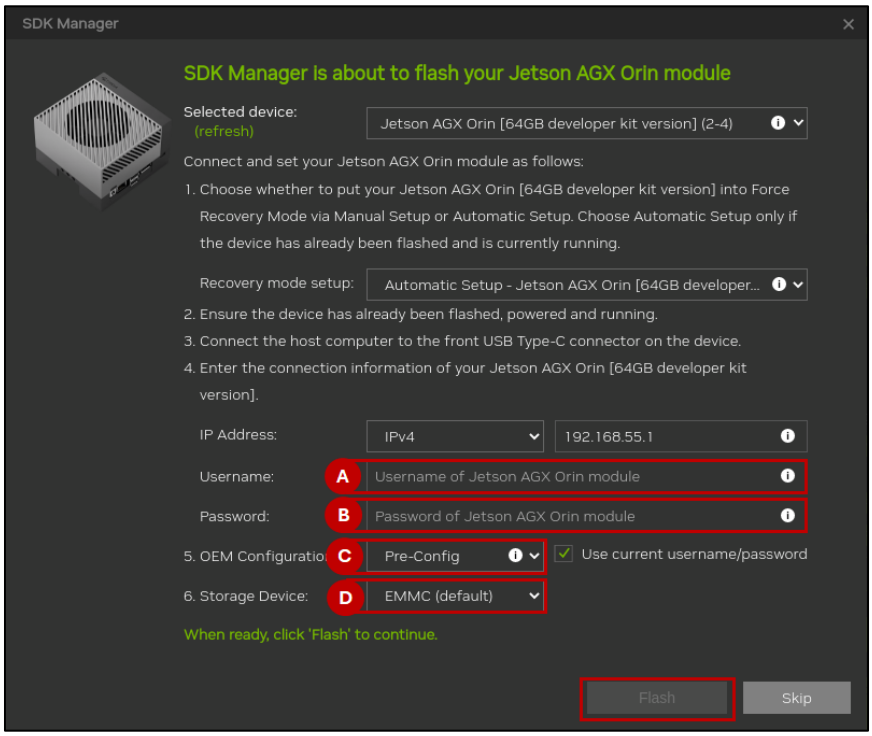


Figure 6.10. SDK Manager

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 6.5. Then proceed to the next section to configure the Jetson AGX Orin itself.

Figure 6.11. Summary Finalization

6.3.1.2. 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 through 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 usermod -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 Avant-X 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 examples[1].

   To isolate that core, edit the /boot/extlinux/extlinux.conf file.
```
sudo apt install nano
sudo apt update
sudo nano /boot/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
```

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

   Note: *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*.

   The command below is shown for demonstrating the use of HOLOLINK_AFFINITY and is not required for system setup.

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

9. 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=nvpmodel.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
```

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

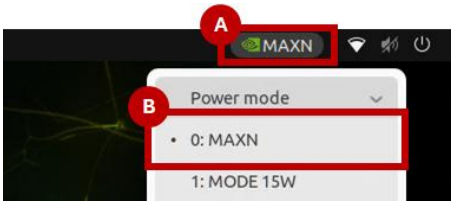


Figure 6.12. MAXN Power Mode

11. Reboot Jetson AGX Orin to apply changes.

```
sudo reboot
```

12. 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=eno1
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
```

13. 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
```

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

```
cat <<EOF | sudo tee /etc/linuxptp/hsb-ptp.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
```

15. 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-ptp.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
```

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.

6.3.1.3. Building the Holoscan Sensor Bridge Demo Container

The MIPI IMX258 camera reference design is based on the HSB release version 2.5.0.

1. Fetch the HSB source code version 2.5.0 from GitHub:
   ```
   git clone https://github.com/nvidia-holoscan/holoscan-sensor-bridge -b 2.5.0
   ```
2. Navigate to the directory at the same level where the `holoscan-sensor-bridge` repository is cloned, copy the patch file **lattice_sensor_bridge_2.5.0.patch** into this directory and apply the patch to support IMX258 camera module.
   ```
   patch -p0 < lattice_sensor_bridge_2.5.0.patch
   ```
3. Verify that **linux_imx258_player.py** exists after applying **lattice_sensor_bridge_2.5.0.patch** to confirm that the patch was applied correctly.
   ```
   ls holoscan-sensor-bridge/examples/linux_imx258_player.py
   ```
   Expected output is as follows:
   ```
   holoscan-sensor-bridge/examples/linux_imx258_player.py
   ```
4. Log in to NVIDIA GPU Cloud (NGC) using the browser on AGX Orin.
   - Register for a developer account for NGC 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.** Run the command *docker login 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
     ```
5. Build the HSB demo container for iGPU (default for AGX Orin).
   ```
   cd holoscan-sensor-bridge
   sh docker/build.sh --igpu
   ```
6. To validate the setup, run the video streaming test on AGX Orin as described in Testing Camera Video Streaming.

6.3.2. Jetson AGX Thor Developer Kit

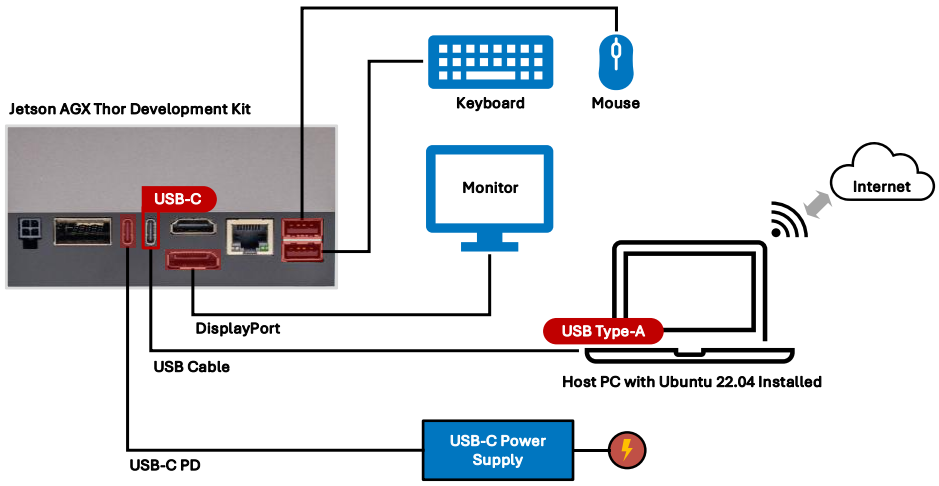


Figure 6.13. Hardware Setup for Flash Jetson AGX Thor

6.3.2.1. Flashing the Jetson AGX Thor using the SDK Manager

1. Ensure the hardware setup is done as shown in Figure 6.13. Also ensure the Ubuntu Host PC is powered up and the internet connection is enabled.
2. Power up the AGX Thor using the USB-C power adapter and make sure the AGX Thor can boot up to the operating system. If the AGX Thor has not been previously flashed, it will not boot up to the operating system and needs to be set to recovery mode.

   Recovery mode step

   a. Ensure the AGX Thor is powered OFF. Verify that the LED indicator is OFF.
   b. Connect the Type-C power cable to the AGX Thor. The power supply must be ON.
   c. Press and hold the Recovery Button (see Figure 6.14 for location).
   d. While holding the Recovery Button, press the Power Button.
   e. Release both buttons simultaneously.

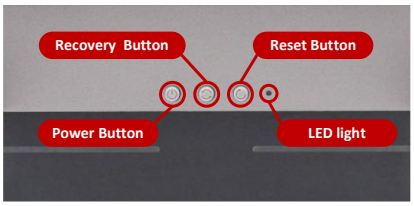


Figure 6.14. AGX Thor Recovery, Reset, and Power Button

Note: Run all commands in this section from the Ubuntu host PC terminal.

3. In the Ubuntu Host PC, open a terminal window and execute the below commands to download 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
```

4. Run the SDK Manager command.
```
sdkmanager
```
5. Sign up for an NVIDIA Developer account at https://developer.nvidia.com if you do not have one.
6. After the SDK Manager is launched, click Login as shown in Figure 6.15. A browser window will open, where you will sign in using your NVIDIA Developer account credentials.
7. Check your email for the authentication verification link and click it to complete the login process. Once verified, the SDK Manager will launch automatically.

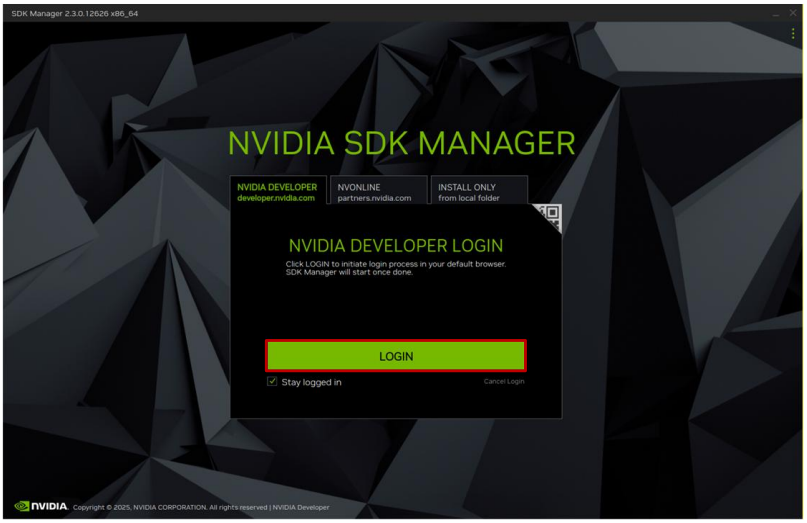


Figure 6.15. SDK Manager Login

8. Select **Jetson AGX Thor Development** as the **Target Hardware** (see letter **A** in Figure 6.16). Verify that the Jetson AGX Thor module is detected. If the module is not detected, repeat the Recovery mode step. Set the SDK version to **JetPack 7.0 (rev. 1)** (see letter **B** in Figure 6.16). Click **CONTINUE** to proceed to **STEP 02 – Details and License**.

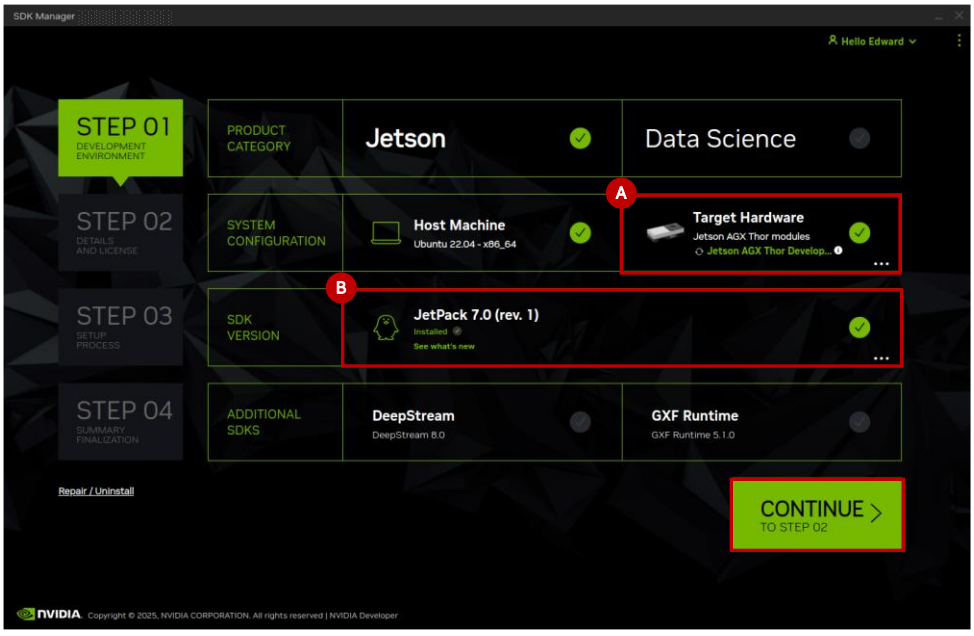


Figure 6.16. STEP 01 – Development Environment

9. In **STEP 02,** select **I accept the terms and conditions of the license agreements** as shown in Figure 6.17. Leave all settings under **TARGET COMPONENTS** as their default values (see letter B). Click **CONTINUE** to proceed with the installation process.

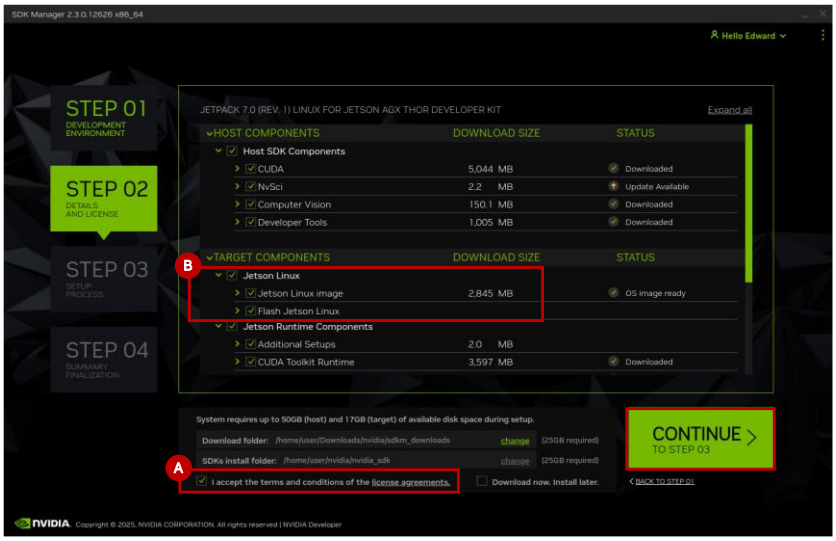


Figure 6.17. STEP 02 – Details and License

10. In the STEP 03 panel, the **SDK Manager** will prompt for a sudo password. Enter the password and click **OK**. The software download and installation process begins. The progress is shown in Figure 6.19.

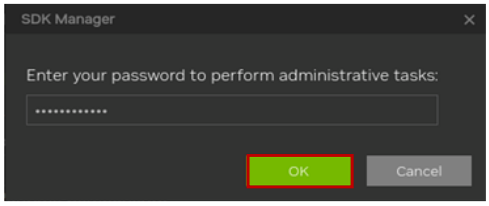


Figure 6.18. Administrative Right Request

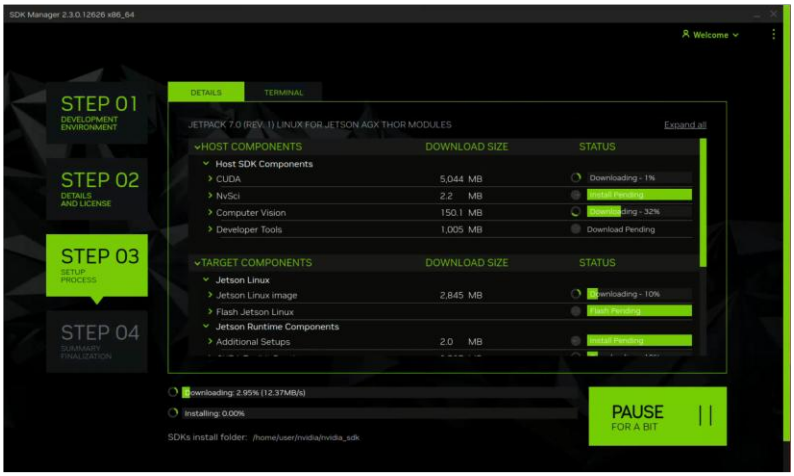


Figure 6.19. STEP 03 – Setup Process

11. After the installation completes, the SDK Manager displays a dialog box to flash the target device. Select **Manual Setup – Jetson AGX Thor Developer Kit** (see letter **A**) in the **Recovery mode setup** shown in Figure 6.20 to begin flashing the target device. Refer to the Recovery mode step for details.

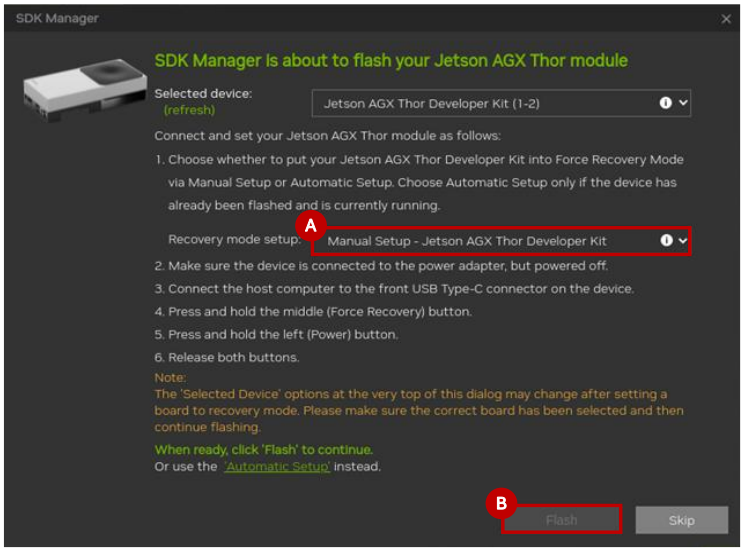


Figure 6.20. Recovery Mode Selection

12. In the setup configuration screen, select **Pre-Config** under the OEM Configuration (letter **A**) as shown in Figure 6.21. Enter the desired **username** (letter **B**) and **password** (letter **C**) to create the new user account in AGX Thor. Leave the **Storage Device** setting at its default value. Click **Flash** to begin the flashing process.

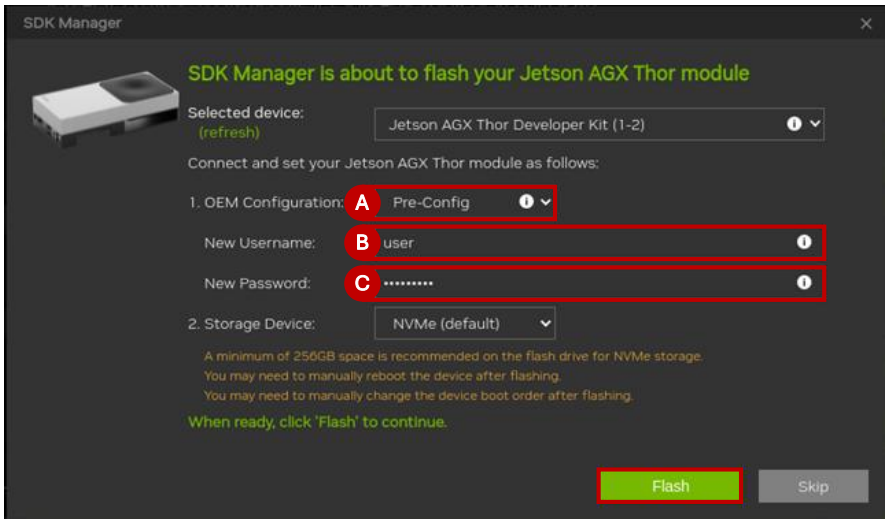


Figure 6.21. Pre-flash Operation Setup

13. After flashing is complete, a full reboot is required (power off, then power on the Jetson AGX Thor, see the power button location in Figure 6.14).
14. Allow the SDK Manager to detect the device and begin the initial setup process. When the AGX Thor reaches the operating system login screen, enter the created **username** at (letter **A**) and **password** (letter **B**) as shown in Figure 6.22. Click **Install** to start installing the SDK component.

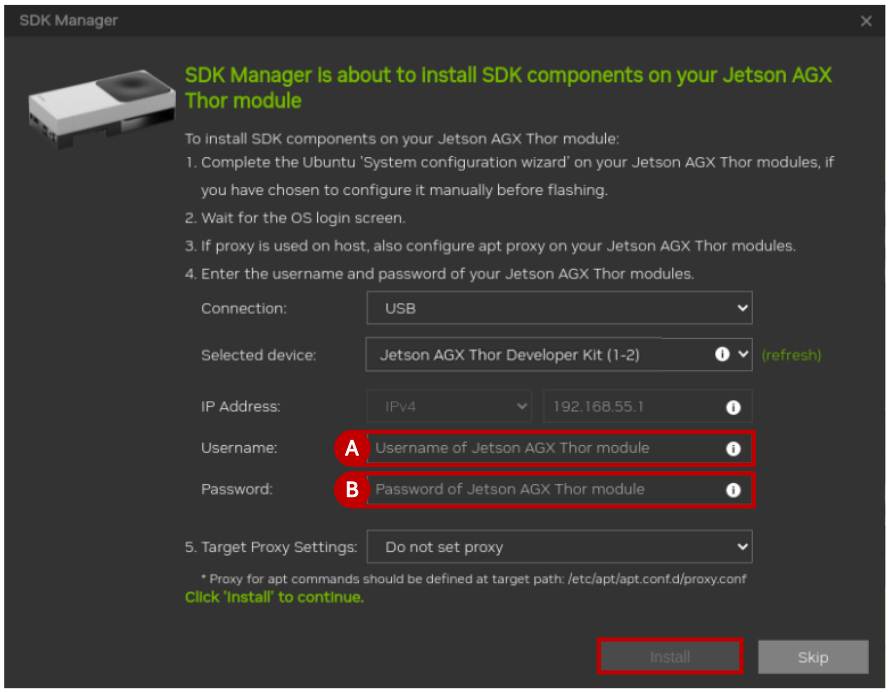


Figure 6.22. Post-flash Installation Dialog

15. After the SDK component installation completes, the **STEP 04 – Summary Finalization** panel appears. Review any warnings or errors displayed in the summary panel. Click **FINISH AND EXIT** to complete the installation process.

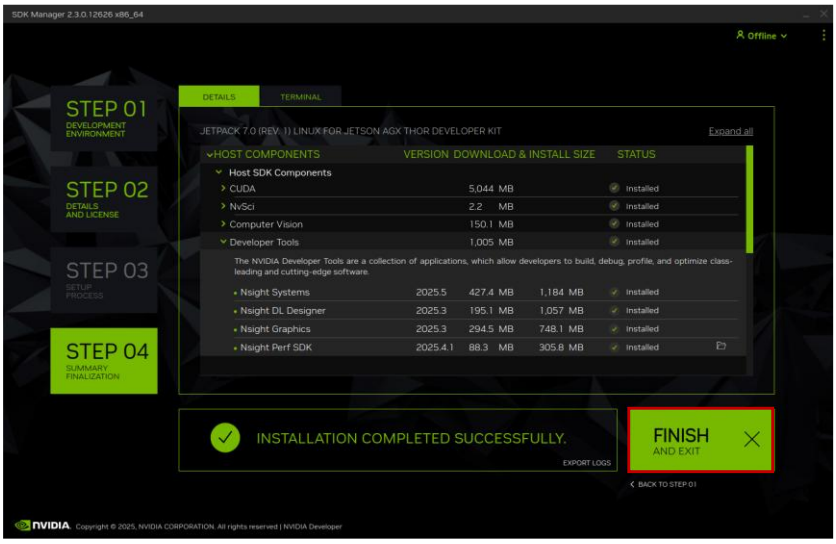


Figure 6.23. STEP 04 – Summary Finalization

16. Select the appropriate Ethernet configuration path.
    a. For 25GbE Ethernet – proceed to the Enabling the 25GbE Ethernet on QSFP Port section then continue to the Configuring the AGX Thor Developer Kit section.
    b. For 10GbE Ethernet – proceed directly to the Configuring the AGX Thor Developer Kit section.

6.3.2.2. Enabling the 25GbE Ethernet on QSFP Port

Note: This section outlines the procedure for enabling the 25GbE Ethernet on the QSFP port. By default, the QSFP port supports only up to 10GbE. Steps 1 to 12 in this section must be executed on the Ubuntu host PC.

1. Install the required prerequisites.
```
sudo apt install git device-tree-compiler
```
2. Verify that all dependencies needed to flash the AGX Thor are installed.
```
cd ~/nvidia/nvidia_sdk/JetPack_7.0_Linux_JETSON_AGX_THOR_DEVKIT/Linux_for_Tegra
sudo tools/l4t_flash_prerequisites.sh
```
3. Navigate to the designated source directory and clone the kernel source.
```
cd
~/nvidia/nvidia_sdk/JetPack_7.0_Linux_JETSON_AGX_THOR_DEVKIT/Linux_for_Tegra/source
sudo ./source_sync.sh -s -k -t jetson_38.2.1 -o $PWD
```
4. Update ownership of the kernel source file as required.
```
sudo chown -R $USER:$USER .
```
5. Navigate to the appropriate directory and copy the *tegra264-p4071-0000.patch* file into it. Apply the patch.
```
cd
~/nvidia/nvidia_sdk/JetPack_7.0_Linux_JETSON_AGX_THOR_DEVKIT/Linux_for_Tegra/source/h
ardware/nvidia/t264/nv-public
patch -p1 -l < tegra264-p4071-0000.patch
```
6. Build the patched DTB:
```
export DTC=/usr/bin/dtc
cd
~/nvidia/nvidia_sdk/JetPack_7.0_Linux_JETSON_AGX_THOR_DEVKIT/Linux_for_Tegra/source
make nvidia-dtbs
```
7. Copy the generated DTB to the following locations:
```
cd
~/nvidia/nvidia_sdk/JetPack_7.0_Linux_JETSON_AGX_THOR_DEVKIT/Linux_for_Tegra/source
sudo cp kernel-devicetree/generic-dts/dtbs/tegra264-p4071-0000+p3834-0008-
nv.dtb ../kernel/dtb/
sudo cp kernel-devicetree/generic-dts/dtbs/tegra264-p4071-0000+p3834-0008-
nv.dtb ../rootfs/boot/
sudo cp kernel-devicetree/generic-dts/dtbs/tegra264-p4071-0000+p3834-0008-
nv.dtb ../bootloader/
```
8. Convert BPMP DTB to a DTS file to enable patching.
```
cd
~/nvidia/nvidia_sdk/JetPack_7.0_Linux_JETSON_AGX_THOR_DEVKIT/Linux_for_Tegra/bootload
er
dtc -O dts -o tegra264-bpmp-3834-0008-4071-xxxx.dts tegra264-bpmp-3834-0008-4071-
xxxx.dtb
```
9. Copy the tegra264-bpmp-3834-0008-4071-xxxx.patch file into the specified directory, apply the patch, and rebuild the BPMP DTB.
```
cd
~/nvidia/nvidia_sdk/JetPack_7.0_Linux_JETSON_AGX_THOR_DEVKIT/Linux_for_Tegra/bootload
er
patch -l < tegra264-bpmp-3834-0008-4071-xxxx.patch
sudo dtc -O dtb -o tegra264-bpmp-3834-0008-4071-xxxx.dtb tegra264-bpmp-3834-0008-
4071-xxxx.dts
```

10. Copy the updated BPMP DTB file to the following location:

```
cd
~/nvidia/nvidia_sdk/JetPack_7.0_Linux_JETSON_AGX_THOR_DEVKIT/Linux_for_Tegra/bootload
er
cp tegra264-bpmp-3834-0008-4071-xxxx.dtb ./generic/
```

11. If necessary, power on the AGX Thor and place the AGX Thor in recovery mode (see Recovery mode **step** above).
12. Flash the AGX Thor with the newly generated kernel and BPMP DTB.

```
cd ~/nvidia/nvidia_sdk/JetPack_7.0_Linux_JETSON_AGX_THOR_DEVKIT/Linux_for_Tegra
sudo ./l4t_initrd_flash.sh jetson-agx-thor-devkit external
```

Note: *Execute Steps 13 and 14 on the AGX Thor.*

13. After flashing completes, power-cycle the AGX Thor and follow the Ubuntu setup wizard to configure the username and password.
14. To verify 25GbE enablement, connect the QSFP 100G-to-4x 25G breakout cable (refer to Figure 6.3) and run the verification command from the terminal.

    Execute the following command:

```
sudo ethtool mgbe0_0
```

    The expected output should confirm that the 25G link is enabled.

```
Settings for mgbe0_0:
   Supported ports: [ TP      MII ]
   Supported link modes:   Not reported
   Supported pause frame use: Symmetric Receive-only
   Supports auto-negotiation: Yes
   Supported FEC modes: Not reported
   Advertised link modes:  Not reported
   Advertised pause frame use: No
   Advertised auto-negotiation: Yes
   Advertised FEC modes: Not reported
   Speed: 25000Mb/s
   Duplex: Full
   Auto-negotiation: on
   Port: MII
   PHYAD: 0
   Transceiver: external
   Supports Wake-on: d
   Wake-on: d
        Current message level: 0x00000000 (0)

   Link detected: yes
```

Note: *Execute Steps 15 to 17 on the Ubuntu Host PC.*

15. Install the SDK components in SDK Manager without flashing the operating system.
16. Launch the SDK Manager and go to the Flashing the Jetson AGX Thor using the SDK Manager section. At *step 9*, unselect all Jetson Linux items (see **Unselect** in Figure 6.24) and click **CONTINUE**.

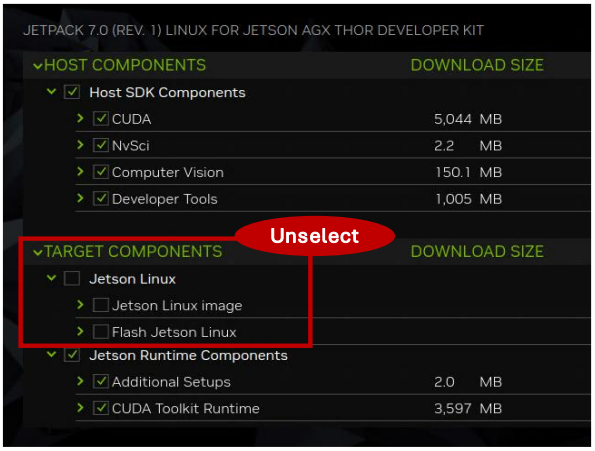


Figure 6.24. STEP 02 – Details and License Target Components

17. Skip the operating system flashing. Do not put AGX Thor into Recovery Mode. Allow it to boot normally until the login screen appears. When the SDK Manager reaches Step 15 (Flashing the Jeston AGX Thor using the SDK Manager section), enter the username and password created in Step 13 of this section. When the installation completes, proceed to Configuring the AGX Thor Developer Kit section.

6.3.2.3. Configuring the AGX Thor Developer Kit

Note: All commands in this section must be executed on the AGX Thor terminal and only need to be performed once.

1. Install the Holoscan SDK.
```
sudo apt update
sudo apt install holoscan=3.7.0-2
```
2. Install the Holoscan Sensor Bridge dependencies.
```
sudo apt install -y git-lfs cmake libfmt-dev libssl-dev libcurlpp-dev libyaml-cpp-dev
libibverbs-dev python3-dev
```
3. Add the user to the Docker subsystem.
```
sudo usermod -aG docker $USER
```
4. Increase the Linux network receive buffer size.
```
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. Isolate a processor core from the Linux kernel when running Linux socket-based examples.
Open **/boot/extlinux/extlinux.conf** for editing.
```
sudo apt update
sudo apt install nano
sudo nano /boot/extlinux/extlinux.conf
```

6. 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
```

   Save and exit by pressing **Ctrl + O**, then Enter, followed by **Ctrl + X**.
   Note: *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. (Complete Steps 1–6 and the procedure in Building the Holoscan Sensor Bridge Demo Container before continuing with Steps 7–12.).*

   The command below is shown for demonstration only and is not required for system setup.

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

7. Enable the network interface (assuming a camera IP address 192.168.0.2).

```
EN0=mgbe0_0
sudo nmcli con add con-name hololink-$EN0 ifname $EN0 type ethernet ip4
192.168.0.101/24
sudo nmcli connection modify hololink-$EN0 +ipv4.routes 192.168.0.2/32
sudo nmcli connection up hololink-$EN0
```

8. Reboot the AGX Thor to apply changes.

```
sudo reboot
```

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

```
sudo apt update && sudo apt install -y linuxptp
EN0=mgbe0_0
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
```

10. Configure the service to run at startup, and start it.

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

11. Run **ptp4l** to send PTP SYNC messages to **$EN0**.

```
cat <<EOF | sudo tee /etc/linuxptp/hsb-ptp.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
```

12. 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-ptp.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
```

6.3.2.4. Building the Holoscan Sensor Bridge Demo Container

Note: All commands in this section must be executed in the AGX Thor.

1. Fetch the Holoscan Sensor Bridge source code from NVIDIA GitHub:
   ```
   git clone https://github.com/nvidia-holoscan/holoscan-sensor-bridge -b 2.5.0
   ```
2. Navigate to the directory at the same level where the `holoscan-sensor-bridge` repository is cloned, copy the patch file **lattice_sensor_bridge_2.5.0.patch** into this directory, and apply the patch to support IMX258 camera module.
   ```
   patch -p0 < lattice_sensor_bridge_2.5.0.patch
   ```
3. Verify that **linux_imx258_player.py** exists after applying **lattice_sensor_bridge_2.5.0.patch** to confirm that the patch was applied correctly:
   ```
   ls holoscan-sensor-bridge/examples/linux_imx258_player.py
   ```
   Expected output is as follows:
   ```
   holoscan-sensor-bridge/examples/linux_imx258_player.py
   ```
4. Log in to NVIDIA GPU Cloud (NGC) using a browser on AGX Thor.
   - Register for a developer account for NGC 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**. Run the command *docker login 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
     ```
5. Rebuild the Holoscan Sensor Bridge demo container using the following command.
   ```
   cd holoscan-sensor-bridge
   sh docker/build.sh --igpu
   ```
6. To validate the setup, run the video streaming test on AGX Thor as described in Testing Camera Video Streaming.

6.4. Testing the System

This section provides details of the reference design testing and bitstream programming on the Avant-X Versa Board with NVIDIA AGX Orin/Thor GPU host connected.

The reference design testing includes the following functionalities:

- Video streaming
  - ISP and MJPEG features
- Body pose estimation
- TAO PeopleNet

6.4.1. Programming the Bit File

This section outlines the steps for uploading the **.bit** file. Currently, the Avant-X Versa Board supports only one method for updating the reference design firmware: a local update through a USB interface. This method allows you to flash the firmware directly to the board using a USB connection, independently of the Holoscan ecosystem. It is ideal for initial setup, debugging, and hands-on development. The following section describes how to upload **.bit** file using the USB method.

6.4.1.1. Local Firmware Update through USB Interface

1. Connect the board to the PC using a mini-USB Type-A cable, as shown in Figure 6.25. Ensure jumper **JP1** is properly connected. Connect the power adapter to the board power port and turn the switch **ON**. The board is now ready to be programmed.

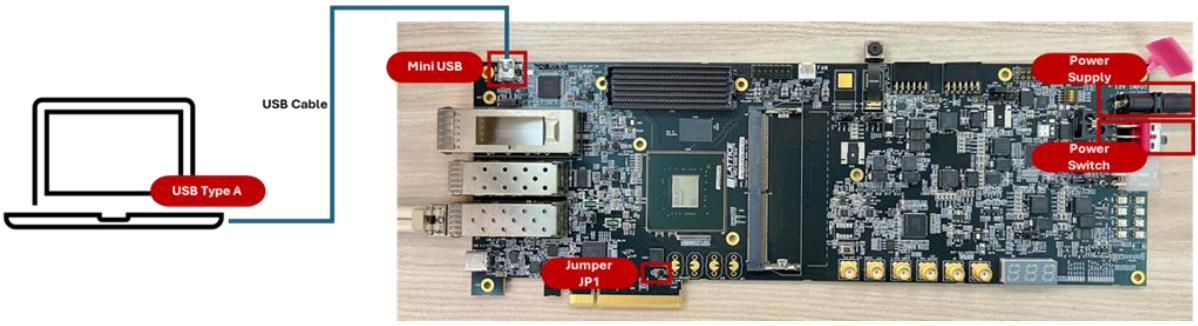


Figure 6.25. Hardware Setup for Bit File Programming

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**.

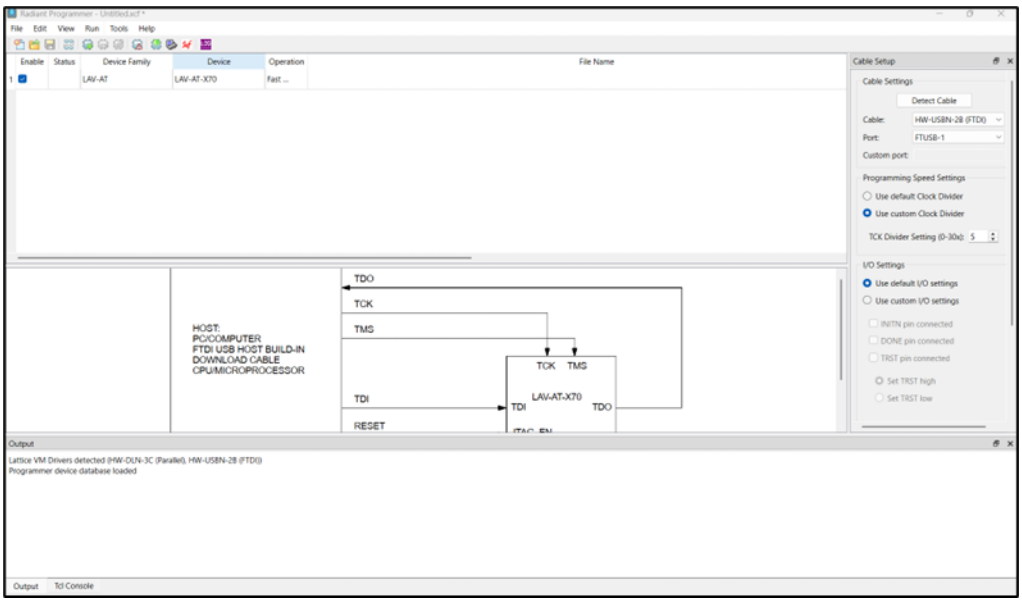


Figure 6.26. Radiant Programmer Window

3. Click **Detect Cable** in the cable setup section, select **FTUSB-1.** Then click **OK** to proceed.

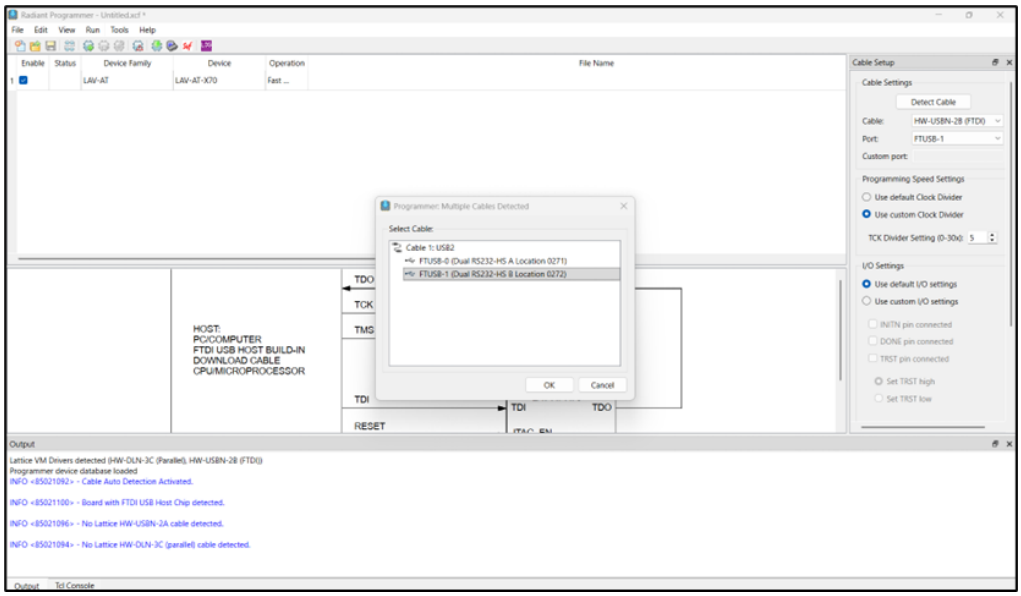


Figure 6.27. Select Cable Settings

4. Select the appropriate SPI flash configuration to flash directly to external SPI flash. Refer to Figure 6.28 for the expected settings. Click the **…** button under the file name bar and choose the bitstream file to program.

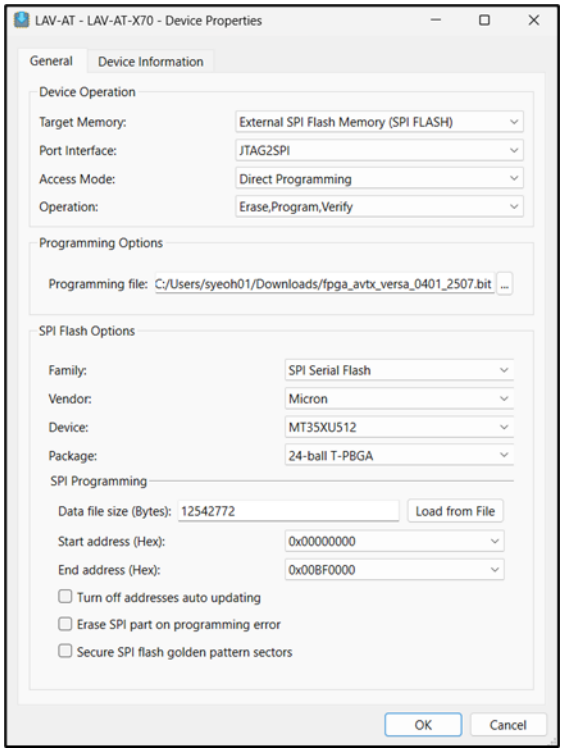


Figure 6.28. Load Bitstream File

5. Click the **Program Device** toolbar icon to load the bitstream into the board, as shown in Figure 6.29.

Figure 6.29. Program Device Toolbar Icon

6. Verify that the programming status is **Operation: successful** in the output window, as shown in Figure 6.30.

   To activate the flashed bitstream, power off the Avant-X Versa Board, then disconnect **JP1** before powering on the Avant-X Versa Board.

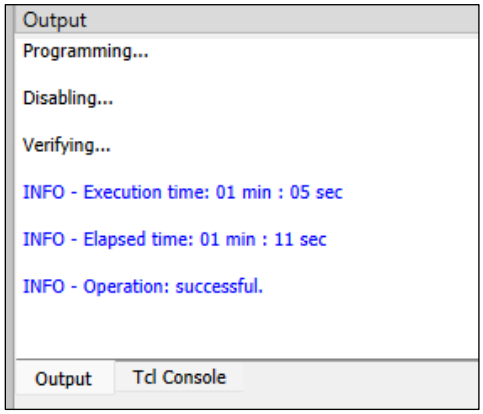


Figure 6.30. Message on Successful Programming

6.4.2. Programming the Golden Bitstream with Jump Table

To support the dual-boot feature on the Avant-X device, you must implement *Feature Row* programming. Feature Row programming is a one-time, irreversible operation. If you intend to enable the dual-boot feature on the Avant-X device, contact Lattice Technical Support.

6.4.3. Testing the Camera Video Streaming

Table 6.2 lists the parameter settings used in the Python scripts for video streaming, body pose estimation, and TAO PeopleNet evaluations. These configurations were applied to all testing activities and are tailored for the IMX258 camera.

Table 6.2. IMX258 Test Command Parameters and Default Values

Parameter	Parameter Value and Description	Default Value
`--runtime-mipi`	**`0`**: Used for fixed lane rate and number MIPI IPK **`1`**: Used for runtime MIPI IPK[1]	1
`--format`	**`raw10`**: Camera streaming data will be in RAW10 format **`mjpeg`**: Camera streaming data will be in MJPEG format **`yuv422`**: Camera streaming data will be in YUV422 format	raw10
`--focus`	Integer value from `-450` to `450` to set the camera focus distance	–200
`--camera-mode`	**`0`**: Set camera mode to 1920x1080 60 FPS **`1`**: Set camera mode to 1920x1080 30 FPS	0

Notes:

1. Do not support Avant-X Versa Board in the current release.
2. The MJPEG format is currently limited to an output frame rate of 30 fps in this release.

Follow these steps:

1. Run the Holoscan Sensor Bridge demo container.
```
cd /<path>/<to>/<holoscan-sensor-bridge>
xhost +
sh docker/demo.sh
```
Note: After running the *demo.sh* script on the terminal, the demo container starts automatically, and the command prompt switches to the container environment.
2. Run the video streaming demo for IMX258 camera. Use the following command.
   - 10GbE or 25GbE Ethernet:
```
python examples/linux_imx258_player.py --runtime-mipi 0
```
Note: Set the camera mode to 0. The 10GbE or 25GbE Ethernet configuration is not supported on the Avant-X devices.

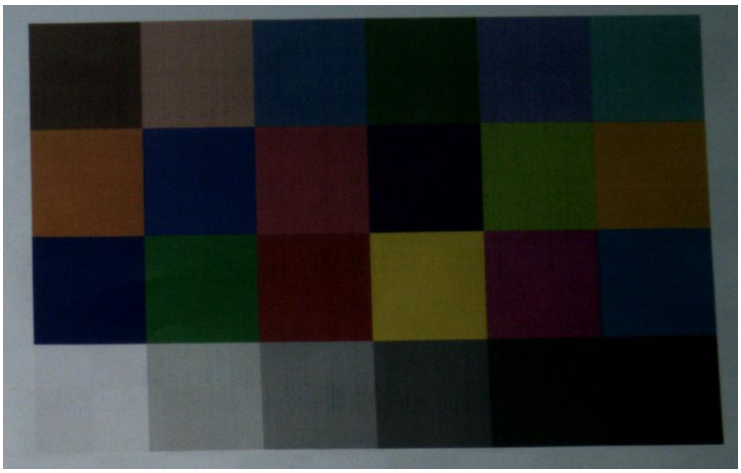
6.4.4. Testing the Camera with ISP and MJPEG Features

In this example, the system is configured using a 10GbE Ethernet interface, with the camera operating at a resolution of 1920 × 1080 at 60 fps. The video streaming function supports three output formats: RAW10, YUV422, and MJPEG. Refer to Table 6.2 for details.

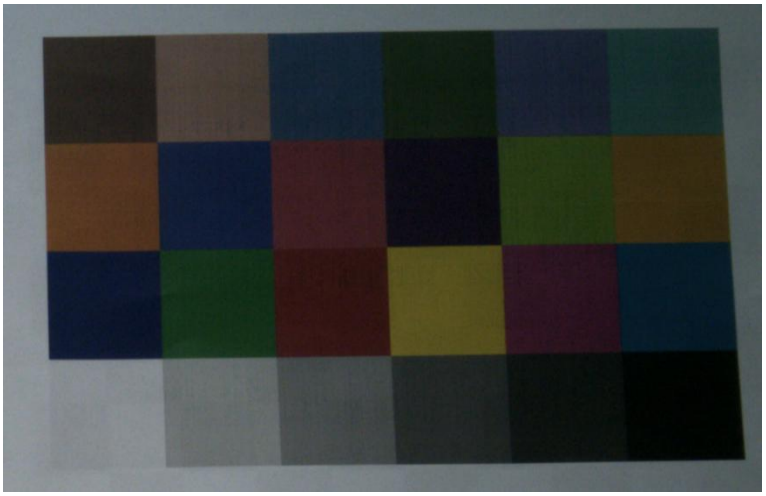
1. Run the Holoscan Sensor Bridge demo container.
```
cd /<path>/<to>/<holoscan-sensor-bridge>
xhost +
sh docker/demo.sh
```
Note: After running the *demo.sh* script on the terminal, the demo container starts automatically, and the command prompt switches to the container environment.
2. Run the video streaming example. To select a streaming format, use the format parameter with mjpeg, raw10, or yuv422. Adjust the lens focus as needed using the focus parameter.
```
python examples/linux_imx258_player.py --runtime-mipi 0 --format mjpeg --focus -200
```

Figure 6.31 shows the image output for different streaming format settings.

RAW10 – Output using built-in NVIDIA ISP and visualizer.



MJPEG – ISP output (YUV422) converts to MJPEG.



YUV422 – ISP enabled output with custom tuning settings for IMX258.

Figure 6.31. Output Image under Different Video Streaming Format

6.4.5. Testing the Camera Body Pose

1. Run the Holoscan Sensor Bridge demo container.

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

Note: After running the *demo.sh* script on the terminal, the demo container starts automatically, and the command prompt switches to the container environment.

2. Complete the one-time prerequisite setup for Body Pose Estimation. Run the following commands inside the container environment.

```
apt-get update && apt-get install -y ffmpeg
pip3 install ultralytics onnx
cd examples
yolo export model=yolov8n-pose.pt format=onnx
trtexec --onnx=yolov8n-pose.onnx --saveEngine=yolov8n-pose.engine.fp32
cd -
```

Note: This setup needs to be performed only once.

3. Run the body pose estimation example to display live pose tracking in the video, as shown in Figure 6.32. You can check the parameter configurations listed in this command:

```
python examples/linux_body_pose_estimation_imx258.py --runtime-mipi 0 --format raw10
```

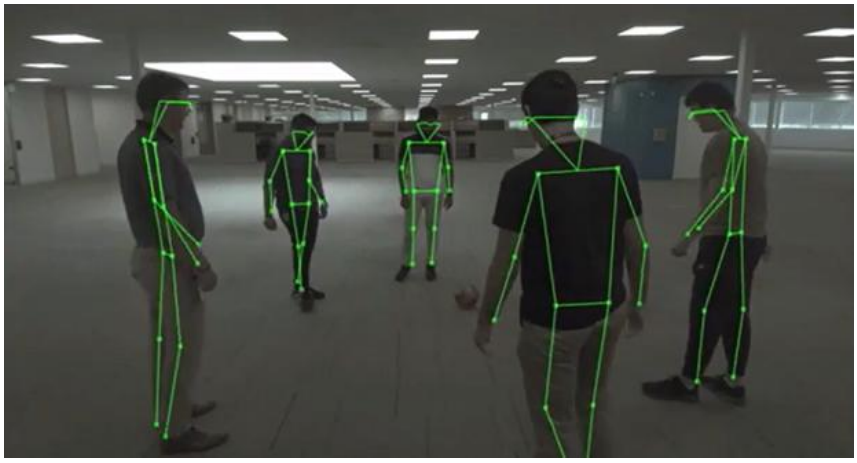


Figure 6.32. Live Body Pose Estimation Output

6.4.6. Testing the Camera TAO PeopleNet

1. Run the Holoscan Sensor Bridge demo container.

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

Note: After running the *demo.sh* script on the terminal, the demo container will start automatically, and the command prompt will switch to the container environment.

2. Complete the one-time prerequisite setup for TAO PeopleNet. Run the following command inside the container environment.

```
wget --content-disposition
'https://api.ngc.nvidia.com/v2/models/org/nvidia/team/tao/peoplenet/pruned_quantized_decrypted
_v2.3.3/files?redirect=true&path=resnet34_peoplenet_int8.onnx' -O
examples/resnet34_peoplenet_int8.onnx
```

Note: This setup only needs to be performed once.

3. Run the TAO PeopleNet example to display live person detection in the video, as shown in Figure 6.33. You can check the parameter configurations listed in:

```
python examples/linux_tao_peoplenet_imx258.py --runtime-mipi 0 --format raw10
```

Note: On the first run, the script may take at least 10 minutes to complete the setup.

Figure 6.33. Live TAO PeopleNet Output

7. Resource Utilization

This section shows the resource utilization of the reference design for the Avant-X (LAV-AT-X70-3LFG1156C) device using the Synplify Pro tool in Lattice Radiant software. 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 Using the Avant-X Versa Board

7.1.1. Resource Utilization with Ethernet 10GbE

Table 7.1 shows one camera sensor interface configured with four MIPI lanes at 720 Mbps, paired with a single-port 10GbE 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 (1x camera sensor interface)	5,671	600	2,088	8,137	36	37.5
Hololink_top (Hololink IP)	15,241	1,386	2,488	22,083	2	23
Ethernet_inst (Ethernet 10GbE host interface)	4,655	0	198	4,021	0	2
Clk_rst	73	0	0	89	0	0
Miscellaneous	288	48	144	787	0	0
Total	25,928	2,034	4,918	35,117	38	62.5

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

Table 7.2 compares the reference design logic consumption with available resources of the Avant-X (LAV-AT-X70-3LFG1156C) device, based on the configuration described in Table 7.1.

Table 7.2. Comparison between the Reference Design Utilization and the Avant-X Device Resources Available

	LUT	PFU Registers	DSP MULT	EBR
Avant-X (LAV-AT-X70) device logic resource	397,440	397,440	1,800	990
Reference design logic resource consumed	32,880	35,117	38	62.5
Reference design logic resource consumed (%)	8.27%	8.84%	2.11%	6.31%

From the place-and-route report, the device SLICE utilization summary after final packing is 14% used (27,344 out of 198,720 slices). The F_{MAX} used is 156.25 MHz.

7.1.2. Resource Utilization with Ethernet 25GbE

Table 7.3 shows one camera sensor interface configured with four MIPI lanes at a lane rate of 720 Mbps, and a single-port 25GbE 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 (1x camera sensor interface)	5,678	600	2,088	8,099	36	37.5
Hololink_top (Hololink IP)	17,682	1,386	2,500	24,104	2	23
Ethernet_inst (Ethernet 25GBe host interface)	10,299	0	272	7,889	0	4
Clk_rst	91	0	10	98	0	0
Miscellaneous	293	48	144	736	0	0
Total	34,043	2,034	5,014	40,926	38	64.5

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 Avant-X (LAV-AT-X70-3LFG1156C) device, based on the configuration described in Table 7.3.

Table 7.4. Comparison between the Reference Design Utilization and the Avant-X Device Resources Available

	LUT	PFU registers	DSP MULT	EBR
Avant-X (LAV-AT-X70) device logic resource	397,440	397,440	1,800	990
Reference design logic resource consumed	41,091	40,926	38	64.5
Reference design logic resource consumed (%)	10.34%	10.29%	2.11%	6.52%

From the place-and-route report, the device SLICE utilization summary after final packing is 17% used (33,117 out of 198,720 slices). The F_{MAX} used is 195.3125 MHz.

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.

8.1. Avant-X Versa Board

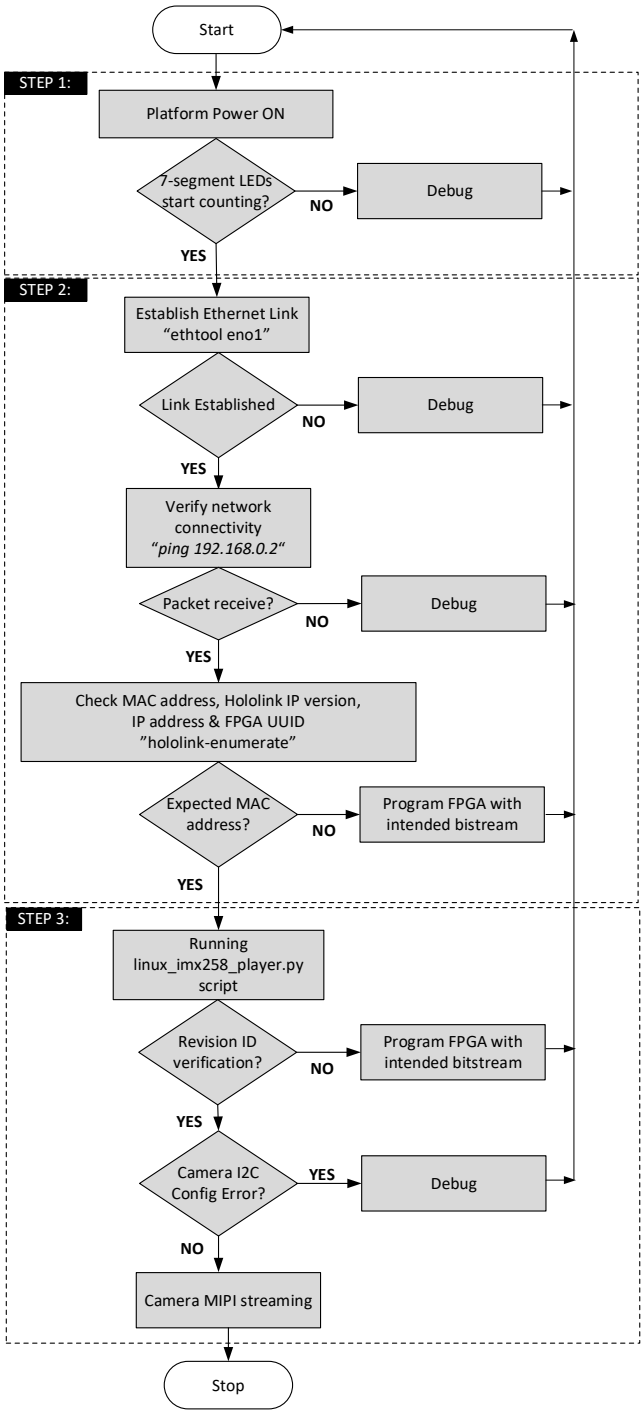


Figure 8.1. Debug Methodology Flow Chart For RD

8.1.1. Powering Up the 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 the 7-segment display.
   After powering up, confirm that the display begins counting. This indicates the Ethernet link is established, the connection is active, and the internal clocks are running correctly.

   Note: This step assumes the HSB Demo container is already set up on the host system.

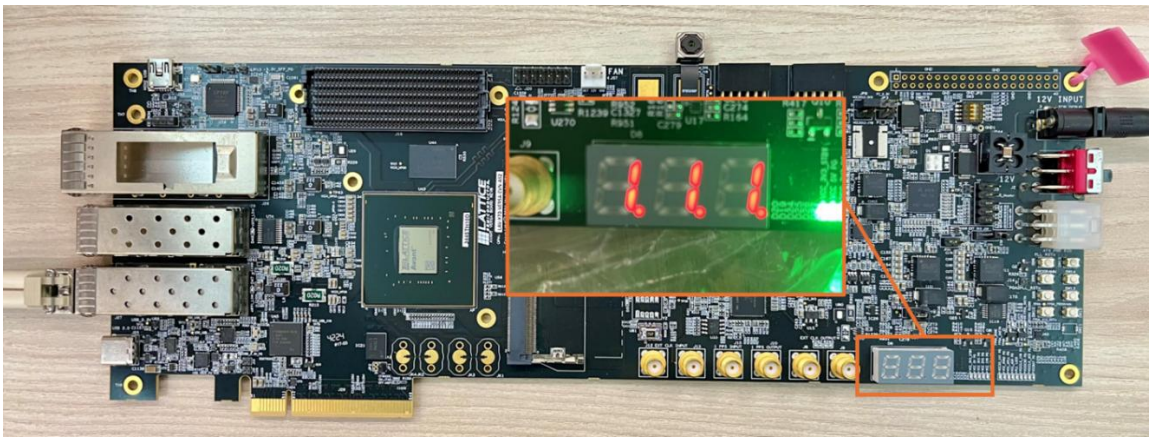


Figure 8.2. 7-segment Display on the Avant-X Versa Board

8.1.1.1. Troubleshooting Common Issues

- **Issue A: The 7-segment display is not on**.
  - Workaround 1: Check the Avant-X Versa Board power supply is connected and the 12 V switch is turned on.
  - Workaround 2: Power 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 Lattice Technical Support.
- **Issue B: The 7-segment display is not counting.**
  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 Lattice Technical Support.

8.1.2. Verifying the Ethernet Connection

1. Check the Ethernet link status.
   On the host terminal, run the *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.

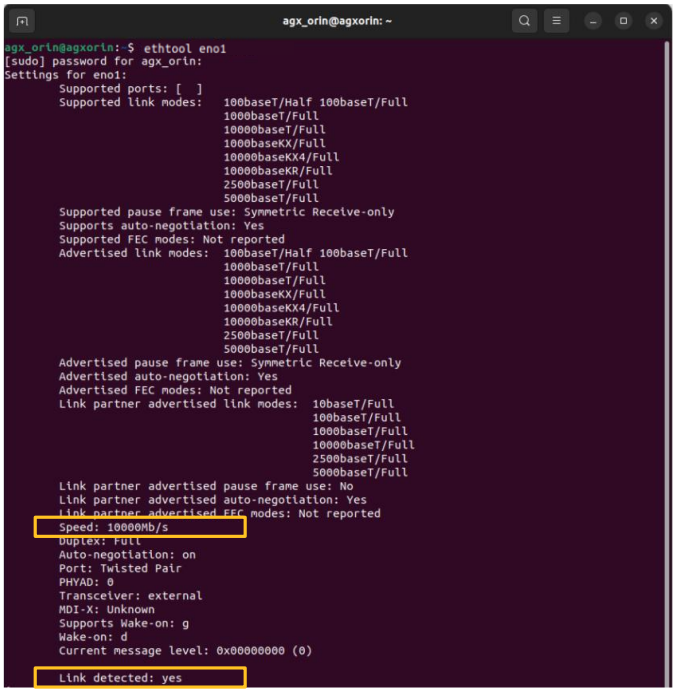


Figure 8.3. Ethernet Link Establishment Status

2. Test the Ethernet connectivity.
   Run the *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 board.

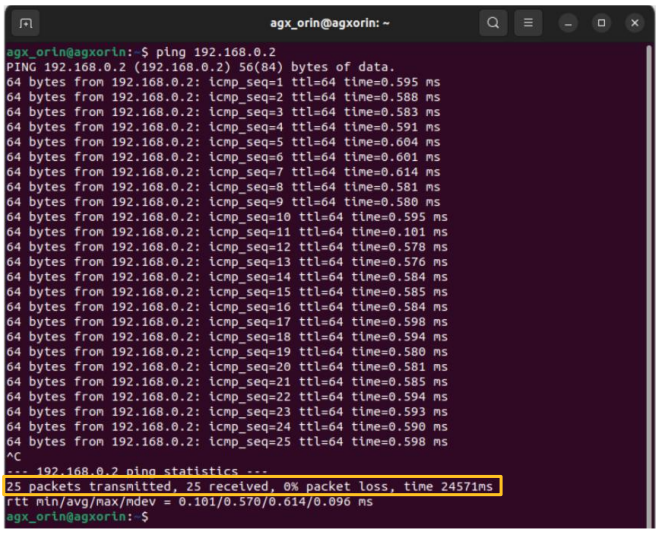


Figure 8.4. Data Packet Received by Host

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.

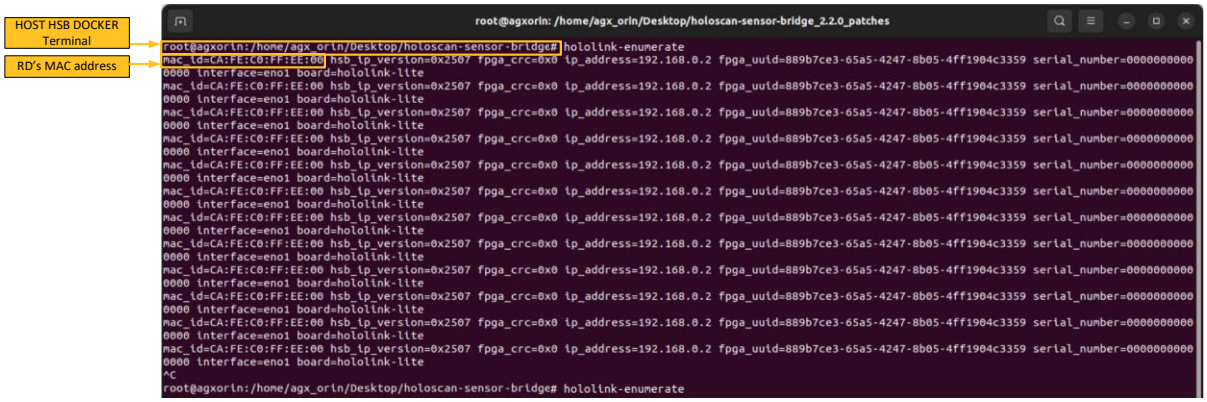


Figure 8.5. Sample Output

8.1.2.1. 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.

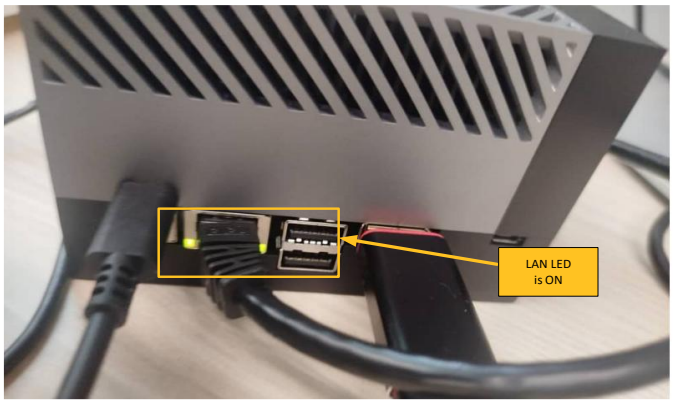


Figure 8.6. LAN Light Indication on Host

8.1.3. Running the Streaming Script (linux_imx258_player.py)

8.1.3.1. Preparing the Streaming Script

Run the script. Follow the steps in the Testing Camera Video Streaming section to run the *linux_imx258_player.py* command on the host HSB demo container prompt.

8.1.3.2. 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 demo container prompt, which represents the bitstream version for the Avant-X device with 10GbE Ethernet.

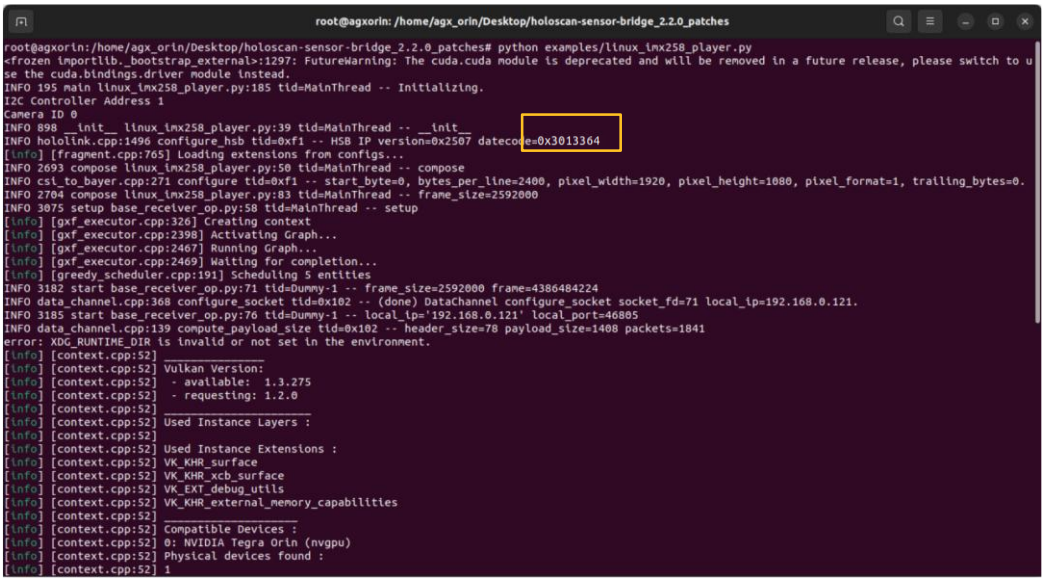


Figure 8.7. Revision ID Version

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.

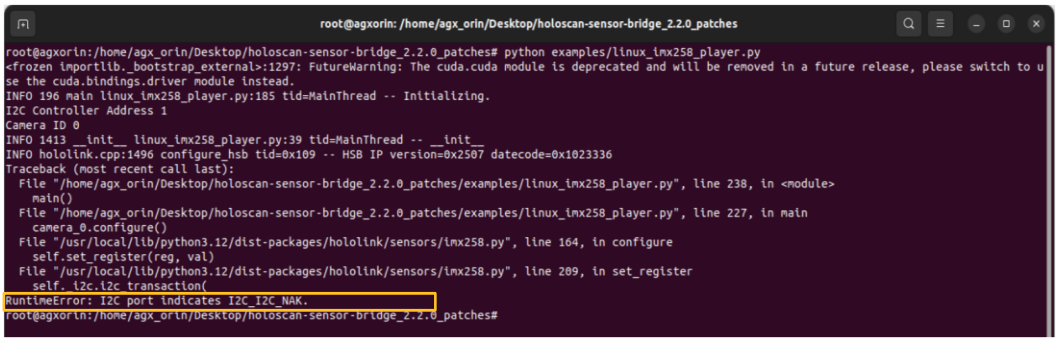


Figure 8.8. Camera I2C Configuration Fail Log

3. Confirm MIPI camera streaming.
   If no errors are detected, the MIPI camera streaming should display successfully on the host monitor.

8.1.3.3. Troubleshooting Common Errors

- **Issue A: Revision ID version mismatch**.
  - Workaround: Reprogram the HSB board with the correct FPGA bitstream version.
- **Issue B: 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.

9. Reference Design Configuration Tool

When customizing the reference design, two files sit at every configuration:

- HOLOLINK_def.svh: SystemVerilog header defining the HDL parameters and DEFINE switches for camera count, MIPI configuration, Ethernet speed, ISP/MJPEG enables, and board identity
- Board .pdc file: Platform Design Constraint file governing pin and placement constraints.

Configuring the HDL Parameter and DEFINE switch section and Configuring the Platform Design Constraint section walk through how to configure these files manually, editing raw text, matching array sizes, and ensuring both files remain consistent with each other. While this approach offers full visibility into every setting, manual errors become more likely, particularly as the number of cameras or Ethernet speed variants grows.

The Reference Design configuration tool addresses exactly that requirement. Rather than opening these files in a text editor, the tool presents the same parameters through a structured graphical interface; enforces valid combinations automatically, and writes both files together in a single *Save* operation, producing the same correctly configured output with far less risk of a manual mistake.

Note: The tool does not launch the Lattice Radiant software or generate bitstreams. After saving the configuration, compile the reference design in the Lattice Radiant software. For details, refer to the Compiling the Reference Design section.

9.1. Installation

Requirements: Windows 10 or 11 (64-bit). Read and write access to the RD project folder.

- Download the Reference Design source zip from the HSB Sensor Interface customization website and extract it fully.
- If the *misc* folder is a separate zip, extract it as well.
- Open the *misc* folder and double-click *rd_config.exe*. No Python or Lattice Radiant software is required.

Caution: Note that the Reference Design configuration tool is designed to work exclusively with its corresponding Reference Design release. Compatibility with other Reference Design versions is not supported.

9.1.1. Windows Defender SmartScreen Warning

When launching *rd_config.exe* for the first time, Windows Defender SmartScreen may display a security prompt. This is expected behavior and does not indicate a problem with the tool.

Steps to Bypass Windows Defender SmartScreen and Launch the Tool

1. On the initial SmartScreen prompt, click **More info**.

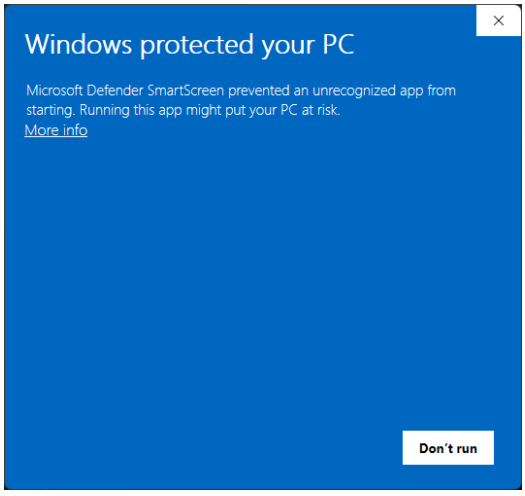


Figure 9.1. Initial SmartScreen Prompt

2. After clicking More info, click Run anyway to launch the tool.

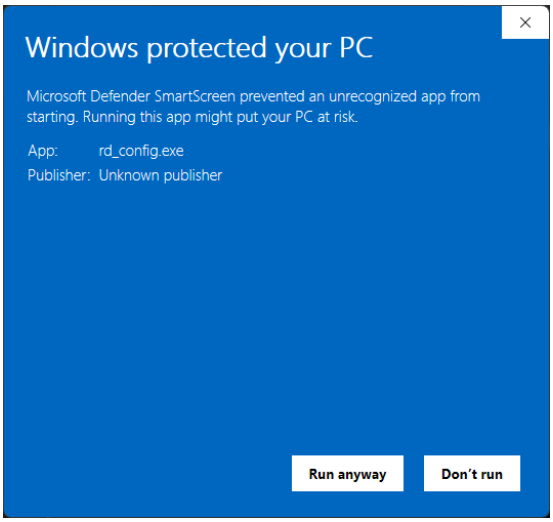


Figure 9.2. Click Run Anyway

Note: This prompt typically appears only on the first launch. If your organization policy prevents running unsigned executables, contact your IT administrator.

9.2. Key Concepts

9.2.1. Preview vs. Save Behavior

Every change in the options pane updates the *HOLOLINK_def.svh* and *board .pdc* preview in the memory only. Project files on the disk are unchanged until you click *Save*. Closing without saving loses all changes.

9.2.2. MIPI Configuration Modes

The Runtime MIPI feature is not supported in Avant; only predefined MIPI modes are available. Changing the Ethernet speed causes the tool to recalculate the maximum supported camera count and valid MIPI lane/lane-rate combinations, while maintaining the currently configured per-camera lane and lane-rate settings. Refer to Table 9.1 for the list of supported MIPI configuration modes.

Table 9.1. Supported MIPI Configuration Modes by Lane Rate

MIPI Lane Rate (Mbps)	Allowed MIPI lanes
720	1, 2, 4
960	2, 4
1440	4
1500	4

Note: Only single camera configurations are supported in the current release.

9.2.3. ISP and MJPEG Dependency

MJPEG requires ISP to be on. Turning ISP off clears MJPEG automatically.

9.3. User Interface

9.3.1. Welcome Screen

When the *rd_config.exe* launches, the welcome screen appears with a large dashed-border drop zone. Drag a project folder or *HOLOLINK_def.svh* into the zone to begin.

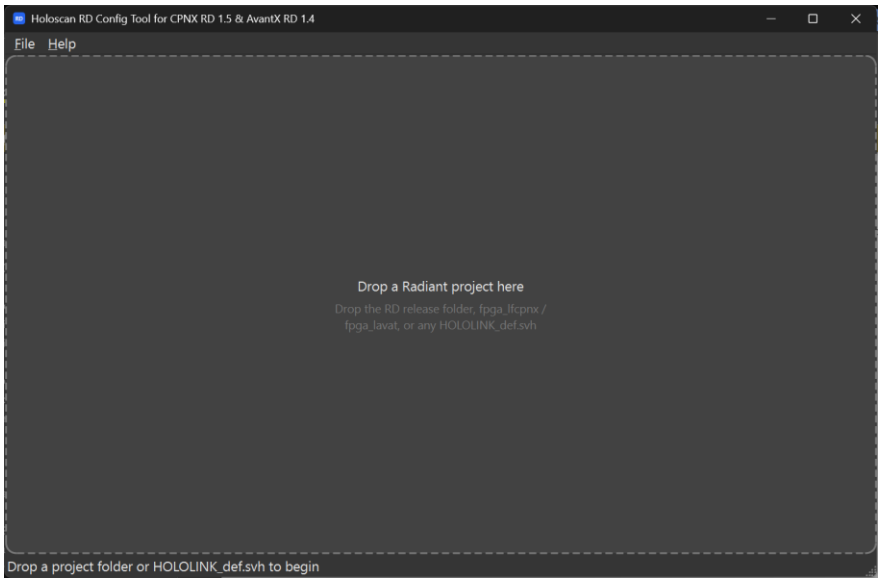


Figure 9.3. Welcome Screen

The drop zone accepts both files and folders. Table 9.2 describes the supported file and folder drop locations.

Table 9.2. Supported File and Folder Drop Location

Drop target	Description
RD project folder	Top-level folder extracted from the RD Source ZIP
<RD project folder>/fpga_lavat	Avant-X FPGA project root
<RD project folder>/fpga_lavat/radiant/src/rtl/hsb_avtx/top/versa_top/HOLOLINK_def.svh	Skips most selection dialogs

9.3.2. Main Workspace

The o*ptions pane* on the left contains all configurable parameters. The *Editor tab* on the right display a live in-memory preview of *HOLOLINK_def.svh* and the *board .pdc* file. The *Save* button is at the top-left of the toolbar.

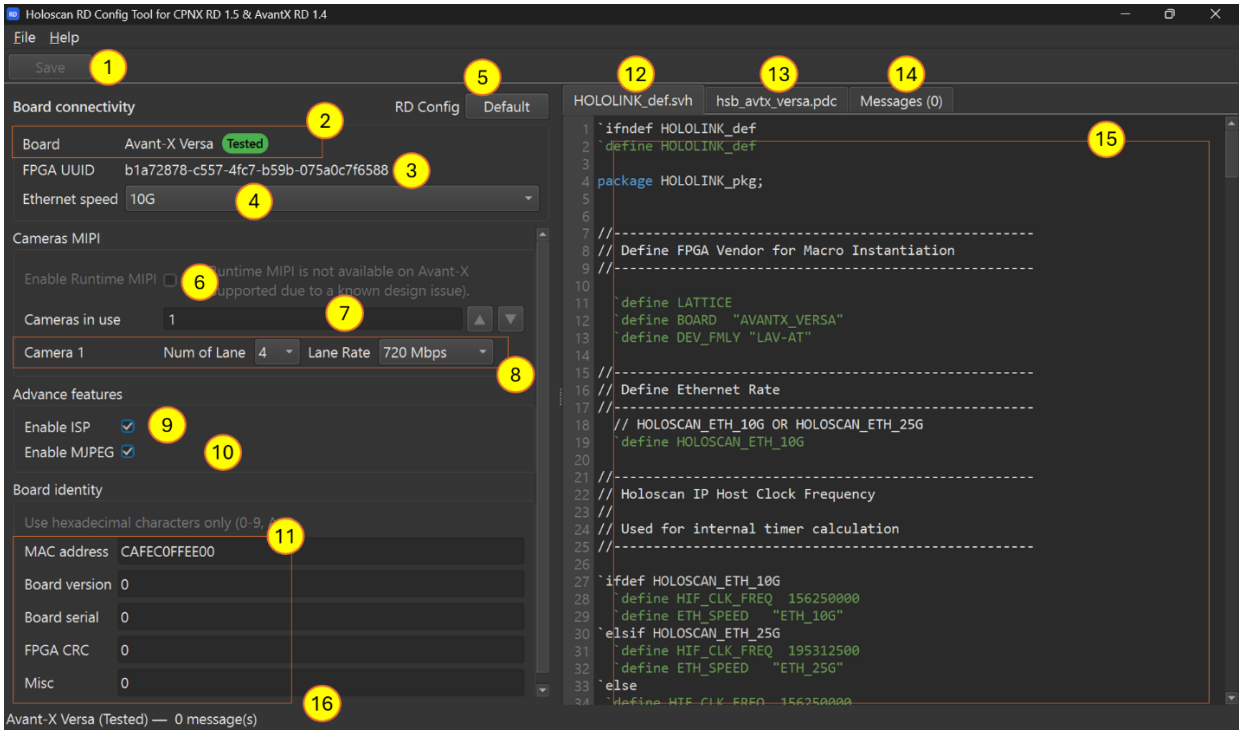


Figure 9.4. Main Workspace – Avant-X Versa Board

The main workspace is described in Table 9.3 through Table 9.7.

Table 9.3. Board connectivity

	Label	Description
1	Save	Commits all in-memory changes to disk. Active only when there are unsaved changes.
2	Board name and badge	Detected board name, read-only. Displays a Tested (green) or Custom (yellow) badge depending on camera count.
3	FPGA UUID	Universally Unique Identifier (UUID) read from HOLOLINK_def.svh — read-only. The UUID differs across Ethernet speed variants (e.g., 10G, 25G).
4	Ethernet speed	Set this first, it drives the Ethernet macro, PDC Ethernet field, and all downstream camera and MIPI limits.
5	RD config/default	Resets all option pane settings to the pack defaults.

Table 9.4. Cameras MIPI

	Label	Description
6	Enable Runtime MIPI	**Not available in this release**. When supported, enabling this option fixes MIPI at 4 lanes / 1500 Mbps and disables the per-camera lane and rate controls. When disabled, lane and rate are configured per camera (Preset MIPI).
7	Cameras in use	Spin control for the number of active camera interfaces.
8	Lane and Rate	Per-camera MIPI lane count and data rate. Valid lanes: 1, 2, 4; Valid rates: 720, 960, 1440, 1500 Mbps (Preset MIPI only).

Table 9.5. Advanced features

	Label	Description
9	Enable ISP	Enables the Image Signal Processing block.
10	Enable MJPEG	Enables Motion JPEG compression. Requires Enable ISP to be on.

Table 9.6. Board identity

	Label	Description
11	Board identity fields	MAC address, Board version, Board serial, FPGA CRC, and misc. Identify the physical board to the host system. Each field must contain valid hex characters only.

Note: All fields accept hexadecimal characters only (0–9, A–F). The tool writes these values to *HOLOLINK_def.svh* on Save.

Table 9.7. Editor tabs and status

	Label	Description
12	SVH preview tab	Live in-memory preview of HOLOLINK_def.svh. Read-only in the tool.
13	PDC tab	Live preview of the board Platform Design Constraint file.
14	Messages tab	Displays validation results. Yellow or red indicator appears when attention is needed.
15	Live SVH preview	Shows the exact text the tool will write on Save, and updates it in real time.
16	Status bar	Shows the current board name, badge, and message count.

9.3.3. Menu Bar and Keyboard Shortcuts

The menu bar provides two menus: **File** and **Help,** covering all project and application actions.

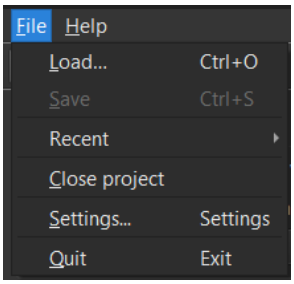


Figure 9.5. File Menu

Table 9.8. Menu Bar Actions and Keyboard Shortcuts

Action	Menu path	Shortcut
Open project	File → Load…	Ctrl+O
Save	File → Save	Ctrl+S
Application settings	File → Settings…	Ctrl+,
Close project	File → Close project	—
Recent projects	File → Recent	—
Version / supported RD	Help → About	—
Quit	File → Quit	—

9.3.3.1. Application Settings

Open through **File → Settings…**. These preferences control how the tool looks and behaves. They do not affect any project files.

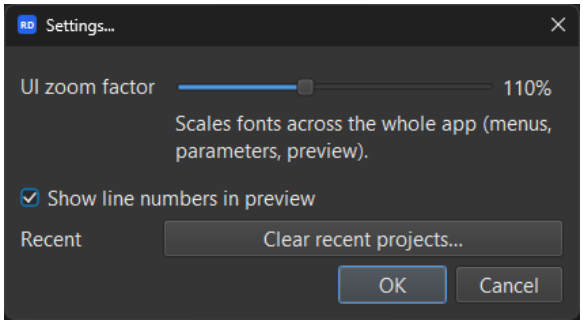


Figure 9.6. Application Settings Dialog Box

Table 9.9. Application Settings Option and Description

Setting	Description
UI zoom factor	Scales fonts across the whole application. Range: 50%–200%.
Show line numbers in preview	Toggles line number display in the SVH and PDC editor tabs.
Recent → Clear recent projects…	Removes all entries from the File → Recent list.

9.3.3.2. About

Open through **Help → About**. Displays the tool name, supported reference design platforms and versions, and a brief description.

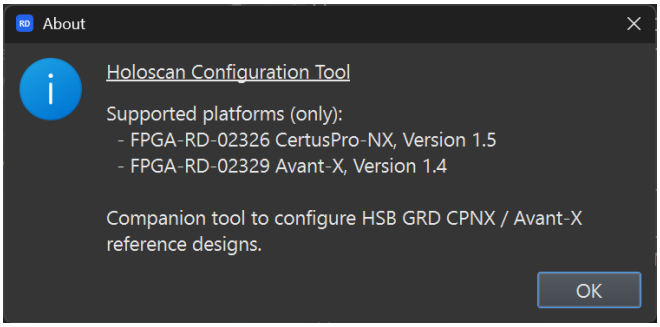


Figure 9.7. About Dialog Box

Table 9.10. About Dialog

Field	Description
Supported platforms	Lists the reference design families and versions the tool supports, example, FPGA-RD-02326 CertusPro-NX v1.5 and FPGA-RD-02329 Avant-X v1.4.
Description	Confirms the tool is a companion for configuring HSB GRD CertusPro-NX / Avant-X reference designs.

9.4. Project Loading

Drag an RD project folder or HOLOLINK_def.svh onto the welcome screen (See Table 9.2 for supported locations), or choose **File → Load…** (Ctrl+O). Since Avant currently supports only a single evaluation board (**Avant-X Versa**), no board-selection dialog is displayed. The tool automatically selects **Avant-X Versa** during project loading.

After the project loads, verify the board name and the FPGA UUID in the options pane. Review the message tab for any warnings or errors that occurred during loading.

Tip: Use **File → Recent** to reopen a recently used project without browsing to the project folder again.

9.5. Configuration Workflow

Follow this order for all Versa boards.

1. Set the Ethernet speed first.

   This drives the Ethernet macro in *HOLOLINK_def.svh*, the PDC Ethernet field, FPGA UUID, and all downstream camera and MIPI limits. Ethernet speed is the root setting, changing it automatically adjusts the camera count limit, ISP/MJPEG availability, and the valid MIPI lane/rate options to match.

2. Set the camera in use.

   The current release supports only a single-camera configuration. The camera count is fixed at one, and the configuration is shown as *tested*.

3. Tune per-camera MIPI (Preset MIPI only)

   Select *Lane* and *Rate* from the dropdowns for each camera. The user-interface filters invalid combinations automatically.

4. Enable ISP/MJPEG is needed.

   Toggle as required. After each change, recheck *Messages*. The ISP and MJPEG reduce the maximum allowed camera count.

5. Check Messages, refer to Table 9.11, then click save (Ctrl+S). After saving, open the project in the Lattice Radiant software to build the bitstream.

Table 9.11. Message Indicators and Recommended Action

Messages state	Action
Clear (no messages)	Proceed to Save
Yellow warning	Review; Save is allowed with confirmation
Red error	Tool prompts Save anyway? Confirm to save with errors, or cancel and fix first

9.6. Board-Specific Guidance

The Reference Design configuration tool supports multiple target platforms. Available configuration options and constraints vary by target platform. Refer to Table 9.12 for details.

Table 9.12. Board-Specific Features and Constraints

Feature	Avant-X Versa
Configuration mode	Full configuration
Ethernet speed	10G or 25G only
MIPI mode	Preset MIPI only
Camera count	More restricted (typically one camera)
ISP / MJPEG	Supported where the UI allows
Save	Enabled
Board badge	Tested

9.7. Save Behavior and Backups

Clicking **Save** writes three files listed in Table 9.13.

Table 9.13. Configuration Data Saved to Each File

File	What changes
HOLOLINK_def.svh	Ethernet macro, MIPI arrays, ISP/MJPEG, board identity
board .pdc	Camera count, per-camera MIPI, Ethernet speed key
.rd_config.json	Board/Reference Design identification metadata

Closing without saving. Click **File → Close project** (or quitting) prompts **Save/Discard/Cancel**. Discard reverts to the on-disk files.

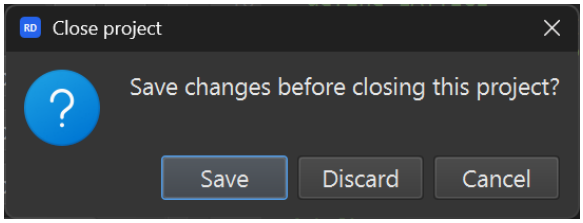


Figure 9.8. Close Project Dialog Box

9.8. Validation Messages

Table 9.14. Message Status Indicator Definitions

Indicator	Meaning
Clear	No issues
Yellow	Warning, for example *Congested Design*
Red	Error, invalid setting

When a Congested Design warning appears in three places: the triangle beside the custom badge, the yellow icon on the message tab, and the warning text in the message panel. Click the *message* to jump to the related line in SVH or PDC.

9.8.1. Diagnostic Codes

Table 9.15. Diagnostic Codes, Causes, and Recommended Fixes

Code	Severity	Cause	Fix
congested_design	Warning	Camera count in Congested Design tier	Reduce the camera count to custom_max or accept and save
preset_config_invalid	Error	Camera/MIPI/ISP/MJPEG combo not in preset matrix	Change one parameter at a time until the combination is valid
runtime_mipi_invalid	Error	Too many cameras for Runtime MIPI tier	Reduce the camera count or switch to Preset MIPI
eth_macro_invalid	Error	Ethernet macro invalid for this board	Re-select a supported Ethernet speed
eth_svh_pdc_mismatch	Error	SVH and PDC disagree on Ethernet speed	Re-select the Ethernet speed to synchronize both files
pdc_eth_invalid	Error	PDC Ethernet value not allowed	Re-select a supported Ethernet speed to update the PDC value

Appendix A. Known Issues and Limitations

1. Remote firmware updates over Ethernet are not supported in the current Avant-X Versa Board customizable HSB Sensor Interface Reference Design.
2. The MIPI D-PHY RX CSI-2 IP is limited to nine available DDRDLL resources. Because each high-speed MIPI lane requires one DDRDLL, a maximum of nine lanes can be supported when operating at 1 Gbps or higher.
3. For MIPI D-PHY RX CSI-2 operation at 1 Gbps or higher, each IP instance must occupy a dedicated I/O bank. Because all required MIPI pins on the Avant X Versa Board are located in Bank 11, only one camera operating at 1 Gbps or higher can be supported.
4. When using a custom design that utilizes more than 30% of available Look-Up Tables (LUTs) , high hold timing failures may occur. Multiple place-and-route iterations may be required. Contact Lattice Technical Support for more information.

References

- Avant-X Versa Board User Guide (FPGA-EB-02063)
- Lattice Avant Multi-Boot User Guide (FPGA-TN-02314)
- CSI-2/DSI D-PHY Receiver IP Core - User Guide (FPGA-IPUG-02081)
- Lattice Radiant Software User Guide
- Avant-X Customizable HSB Sensor Interfaces Reference Design - Source Code
- CSI-2/DSI D-PHY Receiver IP Core web page
- Avant-X Versa Board web page
- Lattice Radiant FPGA design software
- Lattice Solutions IP Cores web page
- Lattice Insights for Lattice Semiconductor training courses and learning plans
- NVIDIA Holoscan Docs Hub
- NVIDIA Jetson Developer Kits
- Information Technology — Digital Compression and Coding of Continuous-tone Still Images — Requirements and Guidelines, ITU-T Recommendation T.81, Sept. 1992. [Online]. Available: w3.org

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 1.4, July 2026

Section	Change Summary
Abbreviations in This Document	Added RD in the list.
Reference Design Configuration Tool	Added this section.

Revision 1.3, June 2026

Section	Change Summary
All	• Updated the Lattice Radiant software version from *2025.2.0.48.0* to *2025.2.1.321.0* globally • Minor editorial fixes.
Abbreviations in This Document	Added the following abbreviations: AOC, AXIS, AWB, CUDA, DCT, DDRDLL, ISP, LSCC, LUT, MJPEG, and QSFP28.

Introduction	• Updated Table 1.1. Summary of Avant-X Versa Board with NVIDIA Jetson AGX Orin.
	  • Added *ISP Lattice Semiconductor Reference Design IP v1.1.1*, *LSCC ISP Builder Linux Orin v1.1.0*, and *MJPEG Lattice Semiconductor Reference Design IP*.
	• Updated Table 1.2. Summary of Avant-X Versa Board with NVIDIA Jetson AGX Thor.
	  • Added *ISP Lattice Semiconductor Reference Design IP v1.1.1*, *LSCC ISP Builder Linux Thor v1.1.0*, and *MJPEG Lattice Semiconductor Reference Design IP*.
	  • Ethernet 25GbE IP version corrected from *v2.2.1* to *v2.2.0*.
	• Expanded the Quick-Start Guide to include separate setup headings for AGX Orin and AGX Thor developer kits.
	• Added new sensor interface feature bullet describing ISP-based image enhancement and MJPEG image decoding support in the Features section.

Directory Structure and File Overview	• Updated all project folder paths to reflect the new directory structure.
	  • RTL files moved from radiant/rtl/ to radiant/src/rtl/
	  • IP files moved from radiant/ips to radiant/ip/
	  • Constraints moved from radiant/pdc/ to radiant/src/constraints/
	  • Bitstream folder renamed from *bitstream/* to *precompiled_file/*
	  • Added new radiant/ip/ISP/ folder for ISP and MJPEG reference design IPs.

Functional Description	• Updated the NVIDIA Hololink documentation reference from the *v2.5.0 release* to the respective *HSB SDK release* in the Overview section.
	• Expanded the Sensor Interface section to add ISP processing and MJPEG compression, increasing the MIPI CSI-2 sensor interface from two to five primary components and adding MJPEG bitstream and ISP YUV422 output formats.
	• Updated Figure 3.5. Overview block diagram of MIPI CSI-2 Sensor Interface to show the new ISP-to-AXIS, MJPEG-to-AXIS, byte-to-pixel, and dual clock-domain CDC paths with MUX-based output selection.
	• Added Table 3.1. Supported Bayer processing and converter steps, Table 3.2. Supported RGB processing and converter steps, and Table 3.3. Supported YUV444 processing and converter steps.
	• Added Multi Camera Frame Synchronization section.
	• Updated the Clocking and Reset Scheme section:
	  • Updated the 10GbE usr_clk from *156.25 MHz* to *322.266 MHz*
	  • Added new clocks to both 10GbE and 25GbE clock tables : OSC_IN_125MHz, eth_refclk_p, and ptp_clk

Customizing the Reference Design	• Updated Figure 4.1. Steps Flow to Customize and Verify the Reference Design to add HSB board selection, run-time MIPI configuration, and ISP or MJPEG enablement steps. • Removed the *Runtime MIPI CSI-2 RX D-PHY IP configuration method* from the MIPI CSI-2 RX D-PHY Ip Generation section. • Added the following in the Generating the IP section:

Section	Change Summary
	• Byte-to-Pixel Conversion IP section • ISP Reference Design IP section • MJPEG Reference Design IP section • Added Figure 4.2. Steps to Generate ISP Lattice Semiconductor Reference Design IP through Figure 4.7. Generate Configuration window illustrating the ISP IP generation and tuning workflow. • Added Table 4.2. Generated ISP IP Folder Structure Description. • Added Table 4.5. ISP or MJPEG Parameters Enablement in Hardware. • Updated the Configuring the HDL Parameter and DEFINE Switch section. • Updated DEFINE file path to reflect the new fpga_lavat/ project structure. • Removed the Runtime MIPI configuration method bullet. • Added image preprocessing (ISP/MJPEG) as a new configurable aspect of the DEFINE switch. • Updated the Configuring the Platform Design Constraint section. • Updated PDC template file paths to reflect the new fpga_lavat/radiant/src/constraints/ directory structure. • Removed the note that restricted Runtime MIPI use to specific MIPI lane rate and lane count values. • Replaced old section *Configuring the MIPI CSI-2 RX D-PHY IP at Runtime* with Enabling ISP or MJPEG at Runtime section. • Updated the Customizing the Reference Design for the Avant-X Versa Board section. • Updated *step 4 cross-reference* to point to the new Building the Holoscan Sensor Bridge Demo Container section, reflecting the consolidated patch and build workflow.
Implementing the Reference Design	• Added Table 6.1. Avant-X Platform and Jetson Module Compatibility at the start of the section, summarizing supported Ethernet speeds for each supported host configuration. • Updated the Jetson AGX ORIN Developer Kit section. • Consolidated the patch application and HSB demo container build steps into a single section workflow for Building the Holoscan Sensor Bridge Demo Container. • Removed the separate *Applying the Patch* subsection. Added NGC login procedure and a validation step referencing the video streaming test. • Updated the Jetson AGX Thor Developer Kit section. • Consolidated the patch application and HSB demo container build steps into a single section workflow for Building the Holoscan Sensor Bridge Demo Container • Removed the separate *Applying the Patch* subsection. Added NGC login procedure and a validation step referencing the video streaming test. • Updated Testing the Camera Video Streaming section to add a parameter reference table (Table 6.2) for the IMX258 streaming demo script. • Added three new application test sections: • Testing the Camera with ISP and MJPEG Features • Testing the Camera with Body Pose • Testing the Camera TAO PeopleNet
Resource Utilization	• Updated all 10GbE resource utilization values to reflect ISP and MJPEG IP additions. Overall SLICE utilization increased from *12%* to *14%*. • Table 7.1. Logic Consumption for Each Instance Available in the Reference Design • Table 7.2. Comparison between the Reference Design Utilization and the Avant-X Device Resources Available • Updated all 25GbE resource utilization values to reflect ISP and MJPEG IP additions. Overall SLICE utilization increased from 11% to 17%. • Table 7.3. Logic Consumption for Each Instance Available in the Reference Design • Table 7.4. Comparison between the Reference Design Utilization and the Avant-X Device Resources Available
Appendix A.	• Updated known issue 2. Corrected the DDRDLL resource count limit for MIPI D-PHY RX CSI-2 IP from *thirteen* to *nine*.

Section	Change Summary
	• Added new known issue 4 warning that custom designs exceeding 30% LUT utilization may experience high hold timing failures and may require multiple place-and-route iterations.
References	• Added the following to this section: • Avant-X Customizable HSB Sensor Interfaces Reference Design - Source Code • ITU-T Recommendation T.81

Revision 1.2, March 2026

Section	Change Summary
All	Changed the document title from *Avant Versa Board – Customizable HSB Sensor Interfaces* to *Customizable HSB Sensor Interfaces on Lattice Avant-X Platforms*.
Abbreviations in This Document	Added CMOS, DDR, Hex, PMA, PROM in this section.
Introduction	Reworked section content.
Directory Structure and File Overview	Reworked section content.
Functional Description	Reworked section content.
Customizing the Reference Design	Reworked section content.
Compiling the Reference Design	• Updated figures in this section. • Updated Table 5.1. Board ID Representation in Hexadecimal.
Implementing the Reference Design	• Updated the hardware requirements section for the Avant-X Versa Board and NVIDIA Jetson AGX Orin. • Removed figure for the *Hardware Setup to support 25G Ethernet*. • Added Programming the Golden Bitstream with Jump Table section.
Resource Utilization	Updated the resource utilization values of the Avant-X Versa Board.
Debug Methodology	Created a subsection for the Avant-X Versa Board.
References	Added the following: • Avant-X Versa Board User Guide (FPGA-EB-02063) • Lattice Avant Multi-Boot User Guide (FPGA-TN-02314)

Revision 1.1, March 2026

Section	Change Summary
All	• Updated Hololink IP version from *2507* to *2511*. • Added Thor platform.

Introduction	• Added the *Thor* platform in this section.
	• Updated the following in Table 1.1. Summary of the Reference Design.
	  • Lattice Radiant software version to *2025.2.0.48.0.*
	  • NVIDIA Holoscan SDK from *3.3.0* to *3.7.0*.
	  • HSB SDK from *2.2.0* to *2.5.0-EA.*
	• Added the following in Table 1.1. Summary of the Reference Design.
	  • Added *Host Software (AGX Thor)*.
	  • Added AGX Thor cable 100G QSFP28 to 4×25G SFP28 Breakout Active Optical Cable (AOC).

Functional Description	Updated Holoscan Sensor Bridge version from *2.2.0* to *2.5.0-EA* in the Hololink IP Overview section.
Customizing the Reference Design	Updated the GitHub link in the Hololink IP section.
Compiling the Reference Design	Updated Lattice Radiant software version to *2025.2.0.48.0* in the Configuring the Lattice Radiant Software to Compile the Reference Design section.

Section	Change Summary
Implementing the Reference Design on the Avant-X Versa Board	• Added hardware and software requirements for NVIDIA Thor. • Updated the HSB SDK version to *2.5.0-EA.* • Added Figure 6.3. Setup with AGX Thor for 10GbE and 25GbE Ethernet. • Added NVIDIA Platform Host Setup section and moved the Jetson AGX Orin Developer Kit under this section. • Added the Jeston AGX Thor Developer Kit section.
Resource Utilization	• Reworked Table 7.1. Logic Consumption for Each Instance Available in the Reference Design and in Table 7.3. Logic Consumption for Each Instance Available in the Reference Design. • Reworked Table 7.2. Comparison between the Reference Design Utilization and the Avant-X Device Resources Available and Table 7.4. Comparison between the Reference Design Utilization and the Avant-X Device Resources.
Appendix A. Known Issues and Limitations	Updated this section.

Revision 1.0, December 2025

Section	Change Summary
All	Initial release.

www.latticesemi.com