CSI-2 to HDMI Reference Design

User Guide

FPGA-UG-02206-1.2

March 2025

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUDANCIES, 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

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
AWB	Automatic White Balance
AXI	Advanced Extensible Interface
B2P	Byte-to-Pixel Converter
CCM	Color Correction Matrix
CFA	Color Filter Array
CMOS	Complementary Metal Oxide Semiconductor
CSI	Camera Serial Interface
DDR	Double Data Rate
DPHY Rx	Receiver part of MIPI DPHY
DRAM	Dynamic Random Access Memory
DUT	Device Under Test
FIFO	First In First Out
FMC	FPGA Mezzanine Connectors
FPGA	Field Programmable Gate Array
FPS	Frames Per Second
GUI	Graphical User Interface
HDMI	High-Definition Multimedia Interface
IP	Intellectual Property
ISP	Image Signal Processing
LED	Light-Emitting Diode
LPDDR4	Low Power Double Data Rate 4
MBPS	Mega Bits Per Second
MIPI	Mobile Industry Processor Interface
PLL	Phase Locked Loop
PPC	Pixels Per Clock
ROM	Read Only Memory
SIM	Simulation
SOF	Start Of Frame
SRAM	Static Random Access Memory
VTG	Video Timing Generator
VTM	Video Timing Merger

1. Introduction

1.1. Overview of the Reference Design

The Lattice CSI-2 to HDMI Reference Design for Lattice Avant™-AT showcases a full video path from a CSI-2 image sensor to HDMI output including sensor image processing in between the interfaces. The reference design uses a CSI-2 image sensor (IMX258) for sensing and Lattice CSI-2 D-PHY Receiver IP, and Lattice Byte-to-Pixel Converter IP to get the pixel data.

The raw video data is then processed using a cascade of basic image signal processing (ISP) modules and sent out in parallel form to a third-party HDMI card. A video timing generator and a frame buffer are used inside the reference design to generate video and timing data suitable for the HDMI device.

The reference design is delivered as a Lattice Radiant™ project which includes a full test environment to perform simulation and the necessary board control logic to run a live demonstration. Both simulation and demonstration systems are built around a core design referred as CSI-2 to HDMI core design in this document. The core design is built using the following Lattice IPs: CSI-2 D-PHY Rx, Byte-to-Pixel Converter, Debayer, Automatic White Balance, and Color Correction Matrix.

1.2. Quick Facts

Table 1.1. CSI-2 to HDMI Reference Design Summary

IP Requirements	Supported FPGA Family	Avant-AT
Resource Utilization	Targeted Devices	LAV-AT-E70ES1
Design Tool Support	Lattice Implementation	Radiant 2024.1.0
	Synthesis	Synopsys Synplify® Pro for Lattice
	Simulation	For a list of supported simulators, see the Lattice Radiant Software user guide.

1.3. Features

Key features of the CSI-2 to HDMI reference design include:

- Full video path from sensor to display
- Simulation environment including self-checking testbench and visual inspection of input/output frames
- Demonstratable on a Lattice hardware board
- Modular design built with Lattice IPs and some additional control logic
- Customizable by user to support additional resolutions and/or sensors

1.4. Conventions

1.4.1. Nomenclature

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

1.4.2. Signal Names

Signal names that end with:

- _n are active low signals
- _i are input signals
- _o are output signals
- _io are bi-directional input/output signals

1.5. Requirements

1.5.1. Hardware

The demonstration system requires the following hardware components:

- Lattice Avant-AT Evaluation Board Revision D (LAV-AT-E70ES1-3LFG1156C)
- Power supply for the evaluation board
- HDMI FMC daughter card (Terasic HDMI FMC (Sil9136-3))
- Lattice Modular FMC adapter (LF-BB-FMC-EVN)
- Image sensor (Sony IMX258)
- HDMI sink (monitor) supporting 1080p at 60 Hz and 4K at 30 Hz, compatible with CTA-861 timing specifications.

1.5.2. Software

The demonstration requires the following software components:

- Lattice Radiant Software version 2024.1.0.34.2
- Radiant Programmer System version 2024.1.0.34.2 software for downloading the FPGA bitstream

1.5.3. Cable

The demonstration system requires the HDMI cable.

1.6. Configurations and Hardware Setup

1.6.1. Supported Configurations

The Radiant project included for the reference design by default is set up for 1080p, 60 Hz video and uses 2 pixels per clock (PPC) data path. To modify the configuration, refer to the Opening the Project section.

Two bitstreams are included in the distribution supporting the following video configurations:

- MIPI_HDMI_RD_1080p.bit: 1080p at 60 Hz
- MIPI_HDMI_RD_4K.bit: 4K at 30 Hz

1.6.2. Hardware Setup

The hardware setup for the demonstration includes the Lattice Avant-AT evaluation board, Lattice Modular FMC adapter card and a HDMI FMC daughter card by Terasic. To assemble the setup, insert the modular FMC adapter into the FMC2 connector, and the HDMI FMC card into the FMC1 connector of the evaluation board as shown in Figure 1.1. Connect the IMX258 sensor ribbon cable to the CN1 connector on the modular FMC adapter. Connect the power supply and the USB mini programming cable to the evaluation board. Connect the HDMI Tx port of Terasic card to a HDMI monitor that can display the selected resolution.

1.6.3. LED and Push Button Descriptions

Six green LEDs on the evaluation board are used in the demo to indicate the status of the I2C configurations of the HDMI device and the image sensor, LPDDR4 initialization and training status, as well as sensor line and frame checker as described in Table 1.2. The LEDs turn off at power on or after reset. The LEDs turn on when the corresponding I2C based configurations are complete.

Table 1.2. LED Descriptions

LED	Description
D6	Image sensor I2C configuration is complete
D7	HDMI I2C configuration is complete
D8	LPDDR4 Memory Controller initialization is complete
D9	Error detected during the LPDDR4 Memory Controller training sequences
D10	Detected an overlap between lines from the image sensor due to insufficient horizontal blanking

LED	Description
D11	Detected an overlap between frames from the image sensor due to insufficient vertical blanking
D12	Reserved – always off
D13	Reserved – always off

A push button is used to reset the design as described in Table 1.3.

Table 1.3. Push Button Descriptions

Push Button	Description
SW1	Active low system reset

1.6.4. Jumper Settings

Jumpers on the board need to be set according to Table 1.4 before powering up the board. The rest of the jumpers should be using the default setting as described in the Avant Evaluation Board User Guide (FPGA-EB-02057).

Table 1.4. Board Jumper Settings

Jumper	Setting	Description
SW7	Set to JTAG position	To program to SRAM directly and load them to the FPGA
JP63	Close	Set VCCIO9 to 1.2 V for the sensor and Rx D-PHY
JP80	Open	

1.6.5. Demonstration Hardware Setup

Figure 1.1 shows the hardware setup of the demonstration.

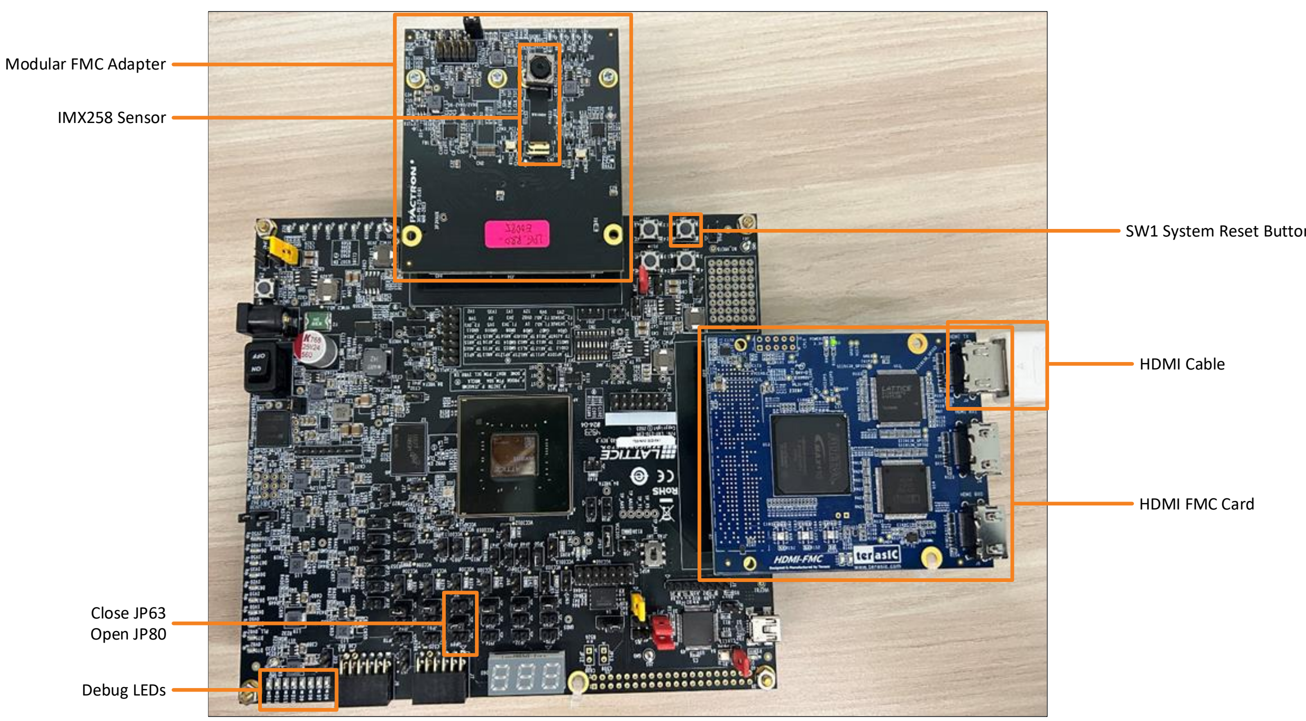

Figure 1.1. Demonstration Hardware Setup

2. Functional Descriptions

The simulation and demonstration designs are both built around the CSI-2 to HDMI core design. This section describes the core design as well as the simulation and demonstration designs.

2.1. CSI-2 to HDMI Core Design

The CSI-2 to HDMI core design, also referred to as MIPI_RX_HDMI_TOP or device under test (DUT) in this document, is shown in Figure 2.1. It uses several Lattice IPs and additional glue logic necessary to create the video data path. The IPs used are CSI-2 D-PHY Rx, Byte-to-Pixel Converter, Debayer, Automatic White Balance, and Color Correction Matrix.

The CSI-2 D-PHY Receiver IP receives the serial data from the image sensor. The Byte-to-Pixel Converter IP converts the output of the D-PHY Receiver IP to pixel data. To help with the timing closure, 2 pixels per clock format is being chosen at this point and onwards. Then, a module called b2p_deb_glue formats the pixel data to AXI-Stream.

The Debayer IP performs color filter array interpolation to obtain RGB components from the RAW input. The Automatic White Balance (AWB) IP corrects the illumination-based color variations in the RGB output of the Debayer IP.

The Color Correction Matrix (CCM) IP that follows the AWB IP does color correction needed to compensate sensor color deviations. The output of the CCM IP is sent to vfb_mc_wrapper, which consist of the Video Frame Buffer IP and LPDDR4 Memory Controller IP. The pixel data is then transferred onto the onboard LPDDR4 DRAM to compensate for inconsistent blanking generated by the sensor.

To be compatible with the chosen HDMI Receiver daughter card in the demo, the pixel data needs to be converted into 1 pixel per clock format. The ppc_2to1 module acts as an interface to receive pixel data (in 2 pixels per clock format) from vfb_mc_wrapper and then converts it into 1 pixel per clock format. Then, the hdmi_sig_gen module transfers the pixel data together with the HDMI timing signals (vsync, hsync, and de) to the HDMI Receiver.

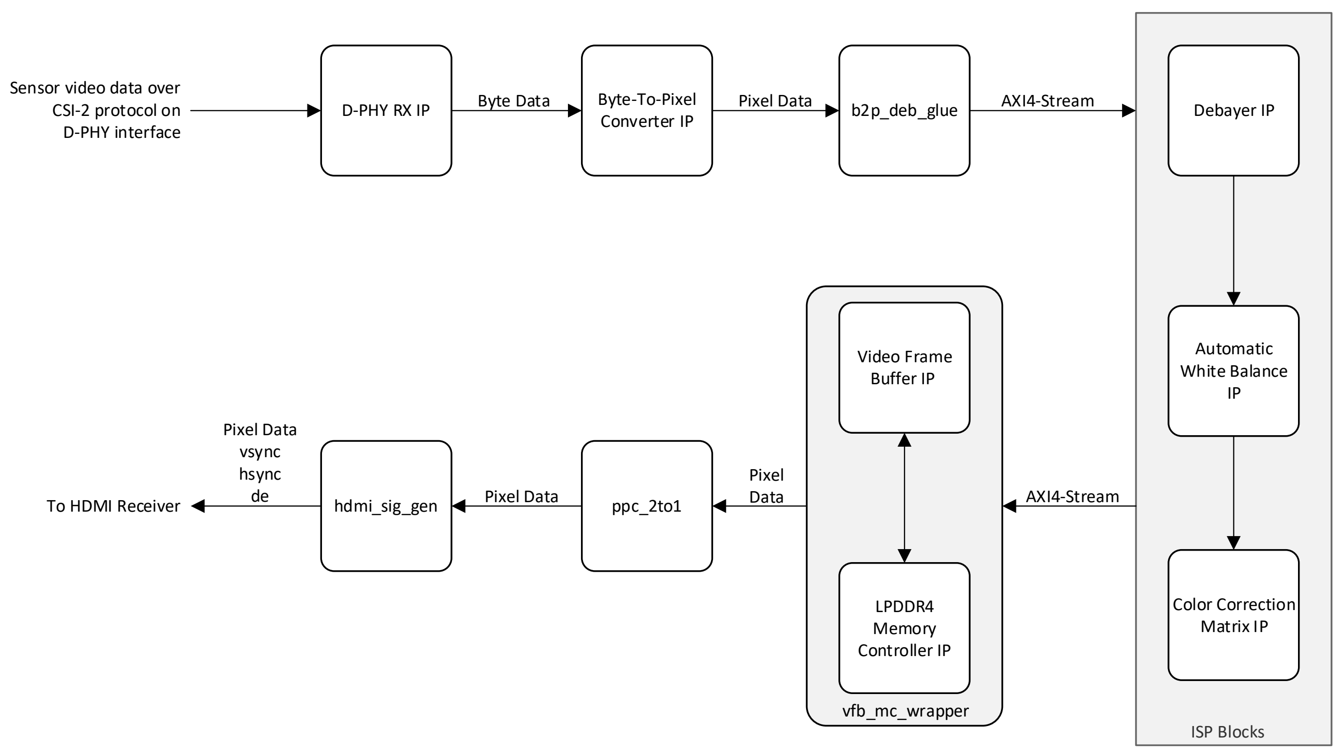

Figure 2.1. CSI-2 to HDMI Core Block Diagram

2.2. Simulation Testbench

The simulation testbench, referred as TB_TOP in this document, includes a data generator and a data checker in addition to the CSI-2 to MIPI core design as shown in Figure 2.2. The data generator reads the Bayerized image data from a text file, converts it to MIPI serial traffic and streams it into the DUT.

The data checker compares the HDMI video data out of DUT with the image data from the golden output file and displays the result in simulation console. In addition to this self-checking testbench, a visual image frame comparison option is also available. To exercise this option, you can input one or more frames of video to the design and render the output image to perform a visual inspection of input and output images. This method is used only for visual equivalency checking, not an accurate byte by byte comparison of input and output data.

Two python scripts are included: one named *png_to_bayer.py* to convert a PNG image file into its equivalent Bayerized data file and another, named *txt_to_png.py* to convert the output text file back to PNG image format for displaying on the monitor. The included scripts only support images in PNG format.

The testbench also allows the capture of simulation waveforms along the pipeline to understand and debug the data flow.

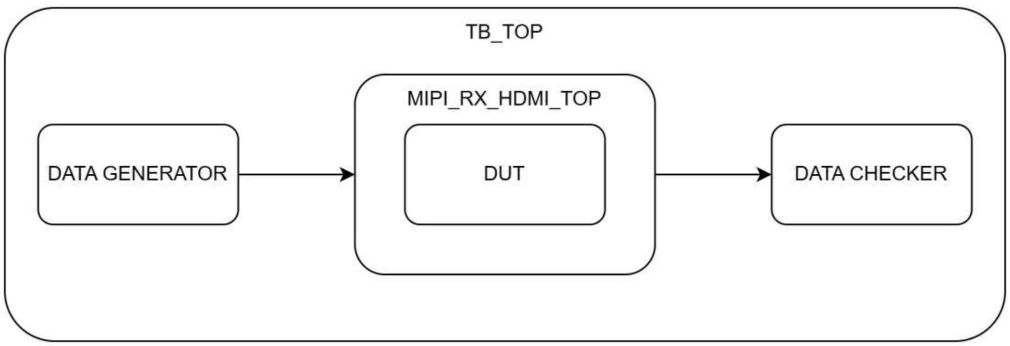

Figure 2.2. CSI-2 to HDMI Testbench (TB_TOP) Block Diagram

While using the simulation testbench, the CSI-2 to HDMI core design replaces the LPDDR4 Memory Controller IP with an emulated AXI LPDDR4 model to speed up the simulation.

2.3. Demonstration Design

The demonstration design, also referred to as FPGA_TOP in this document, showcases an end-to-end video path from the image sensor to HDMI output using Lattice hardware boards. The demonstration design consists of the CSI-2 to HDMI core design and other modules to control the clocks, sensor and HDMI device in the hardware setup as shown in Figure 2.3. The hardware setup used for the demonstration is shown in Figure 1.1. The following sections describe the major components of the demonstration design.

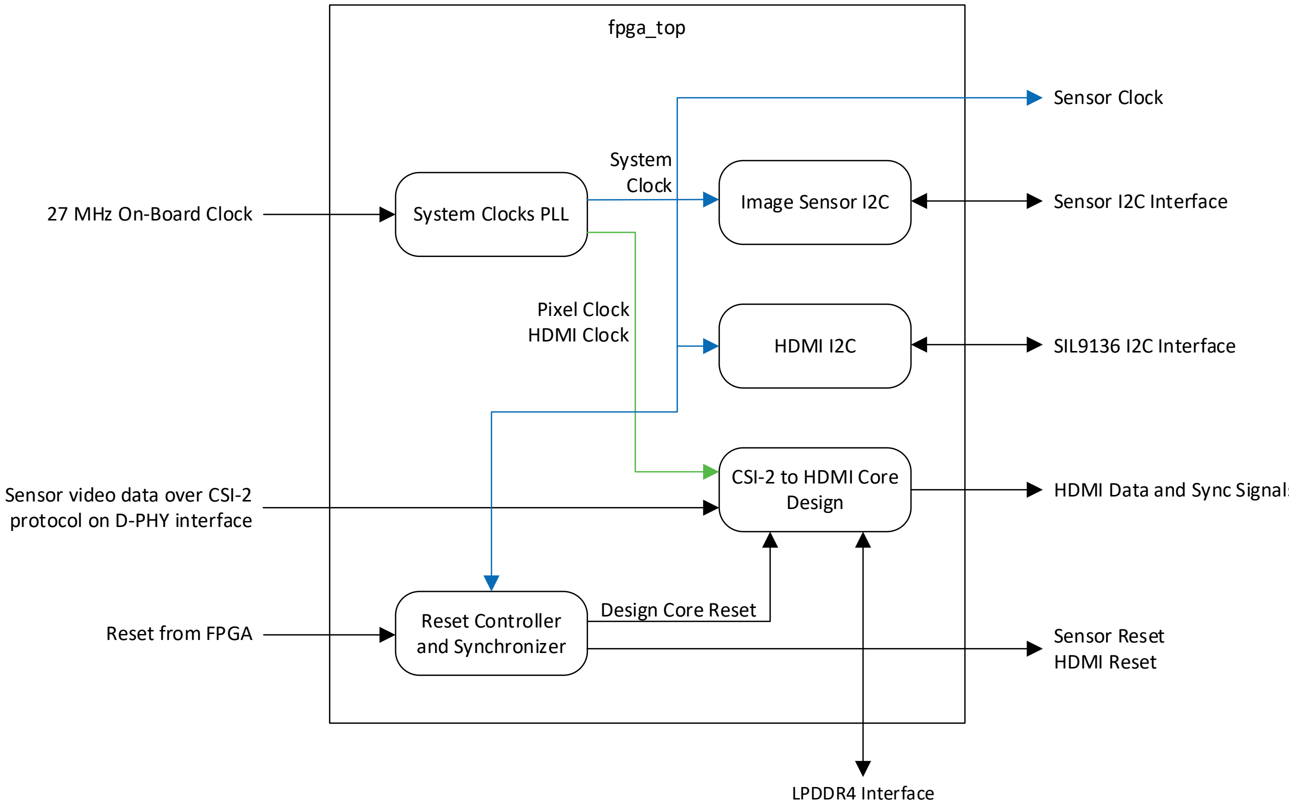

Figure 2.3. CSI-2 to HDMI Demonstration Design Block Diagram

2.3.1. System Clocks PLL

This PLL generates the synchronous clock input for the MIPI CSI-2 D-PHY Receiver as well as the clocks required for the IMX258 image sensor and I2C modules. It directly takes the incoming clock from the board as input to generate the clocks. This PLL also generates the pixel clock and HDMI clock.

2.3.2. Reset Controller and Synchronizer

This module generates the reset signals for the DUT, both I2C controllers, the image sensor, and the HDMI transmitter. It takes the clock and reset incoming from the board as inputs.

2.3.3. HDMI I2C

This module sends the I2C commands required to appropriately configure the HDMI transmitter device based on the design operating resolution. The commands are read from a ROM instantiated inside the design.

2.3.4. Image Sensor I2C

This module sends the I2C commands required to appropriately configure the IMX258 image sensor based on the design operating resolution. The commands are read from a ROM instantiated inside the design.

3. IP and Signal Descriptions

The attributes and signal description of the IPs used in DUT and FPGA_TOP are described in the following sections.

3.1. IP Descriptions

As described in Functional Descriptions section, the design uses multiple Lattice IPs including MIPI CSI-2 D-PHY Receiver, Byte-to-Pixel Converter, Debayer, Automatic White Balance, Color Correction Matrix, Frame Buffer, LPDDR4 Memory Controller, PLLs, and ROM.

The parameters of the IPs used in the provided design (1080p video path design) are shown under *Value for 1080p Configuration (Default)* column in the IP Attributes tables. The parameters for the 4K video path are shown in the last column in those tables.

Design considerations for parametrization of different IPs:

- Make sure all the IPs are updated and have minimum versions as shown in the screenshots of each IP GUI.
- Make sure the changes in gear, number of serial data lanes, data type (bits per pixel), PPC, frame resolution, operating frequencies, and data rates are reflected in the GUIs of respective regenerated IPs.
- Set MIPI CSI-2 D-PHY IP link rate to match the image sensor data rate.
- Set the B2P clock frequencies according to the data type and PPC used so that the input and output data rates match the MIPI CSI-2 D-PHY IP link rate.
- Since the complete design resolution (for both simulation and implementation) is dependent on B2P IP word count parameter, set it to one of the two values i.e. 4800 for 4K resolution and 2400 for 1080p resolution through its GUI.
- The parameters of the CCM IP and AWB IP must match that of the Debayer IP.
- All the IPs (except Memory Controller IP) need to be regenerated when the project is opened for the first time. In addition, all the ROM IPs need to be regenerated every time there is a change in the contents of the memory (.mem) file.
- Set the clock frequencies for each IP according to the data type, PPC, frame resolution, and frame rate used.
- Modify the operating clocks and IPs based on the operating resolution of the whole design.

3.1.1. MIPI CSI-2 D-PHY Receiver IP

MIPI CSI-2 D-PHY Receiver IP for Lattice Avant-AT FPGA converts CSI-2 serial data to 32-bit parallel data. It uses 4-serial data lanes, has 8 gearings, and works in high-speed clock mode. It is implemented as a soft IP on the FPGA. It also generates an output byte clock based on the data type which is passed on to the top-level logic of the DUT. A screen shot of the IP GUI is shown in Figure 3.1 and the attributes in Table 3.1. For more information, refer to the CSI-2/DSI D-PHY Rx IP User Guide (FPGA-IPUG-02081).

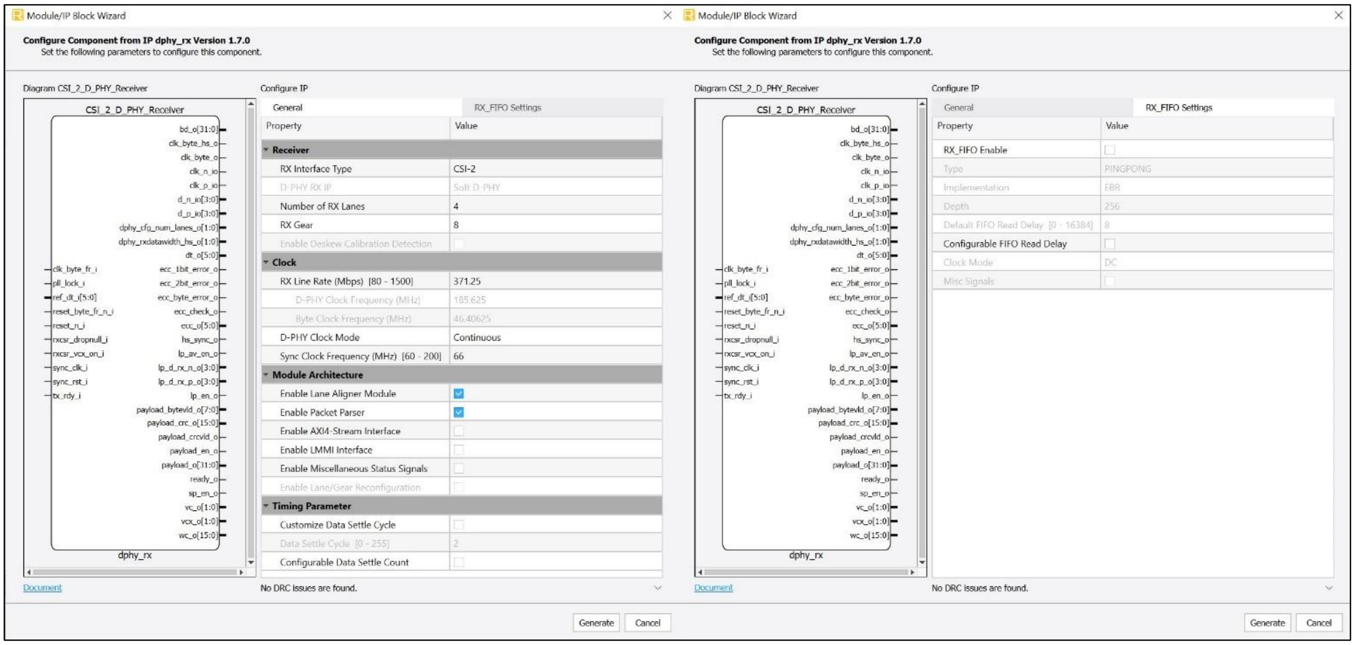

Figure 3.1. CSI-2 to HDMI Demonstration Design Block Diagram

Table 3.1. MIPI CSI-2 D-PHY IP Attributes

Attribute	Selectable Values	Value for 1080p Configuration (Default)	Value for 4K Configuration
General			
Receiver			
Rx Protocol Interface	CSI-2, DSI	CSI-2	CSI-2
Number of D-PHY Data Lanes	1, 2, 3, 4	4	4
Rx Gear	8, 16	8	8
Module Architecture			
Enable Lane Aligner Module	Enabled, Disabled	Enabled	Enabled
Enable Packet Parser	Enabled, Disabled	Enabled	Enabled
Enable AXI-4 Stream Interface	Enabled, Disabled	Disabled	Disabled
Enable LMMI Interface	Enabled, Disabled	Disabled	Disabled
Enable Miscellaneous Status Signals	Enabled, Disabled	Disabled	Disabled
Clock			
Rx Line Rate (MBPS)	80–1800	371.25	742.5
D-PHY Clock Mode	Continuous, Non-Continuous	Continuous	Continuous
Sync Clock (MHz)	60–200	66	66
Timing Parameter			
Customize Data Settle Cycle	Enabled, Disabled	Disabled	Disabled
Data Settle Cycle	1–27	4	4
RX_FIFO Settings			
RX_FIFO Enable	Enabled, Disabled	Disabled	Disabled
Misc Signals	Enabled, Disabled	Disable	Disable

3.1.2. Byte-to-Pixel Converter IP

Byte-to-Pixel Converter IP converts CSI-2 standard-based video payload packets from D-PHY Receiver Module output to pixel format. In addition, Byte-to-Pixel Converter IP generates image sensor control signals in the pixel domain, based on the CSI-2 synchronization packets. The configuration GUI of Byte-to-Pixel IP is shown in Figure 3.2 and the attributes are shown in Table 3.2. For more information, refer to the Byte-to-Pixel Converter IP User Guide (FPGA-IPUG-02079).

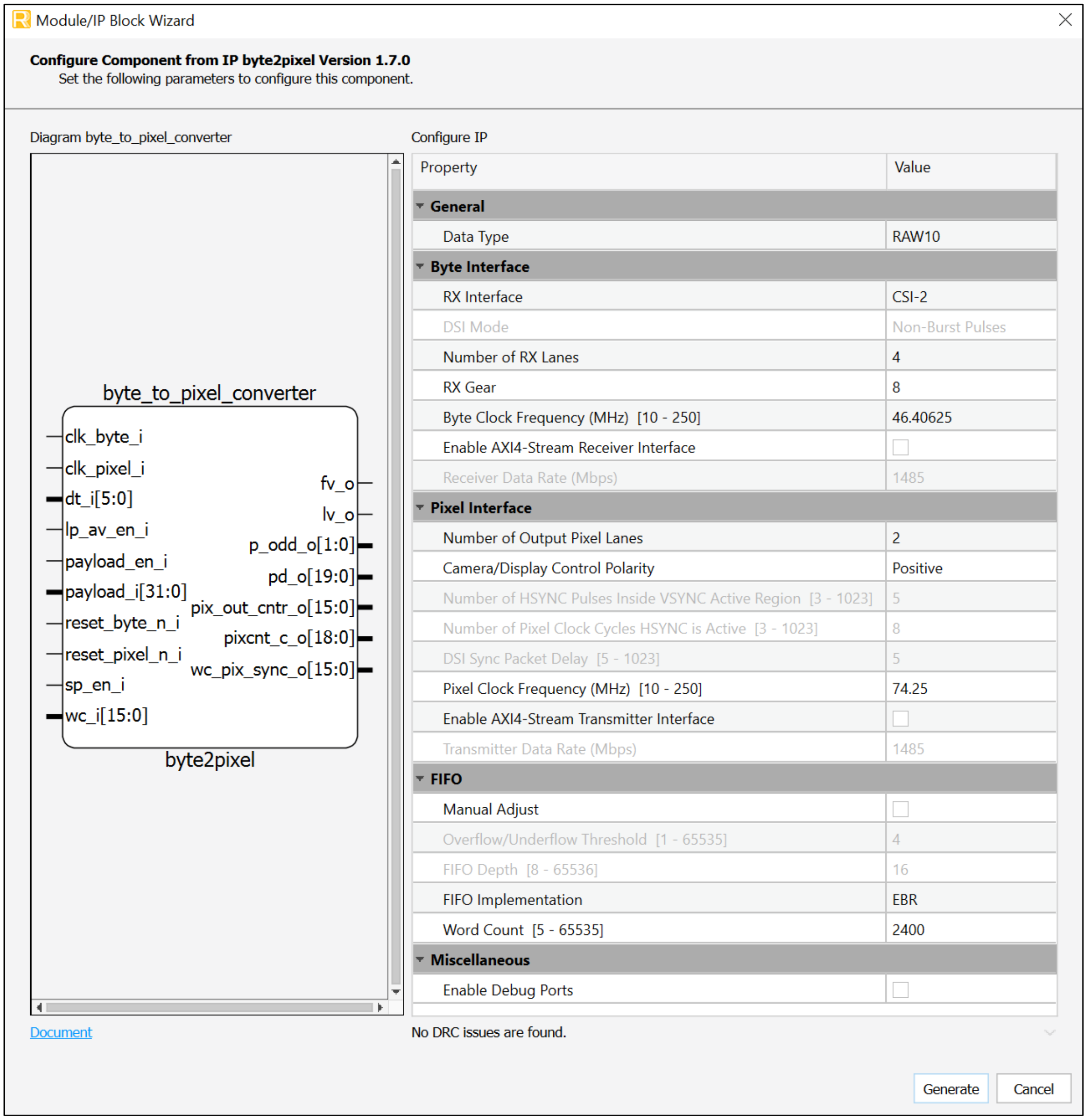

Figure 3.2. Module/IP Block Wizard of Byte-to-Pixel Converter IP

Table 3.2. Byte-to-Pixel Converter IP Attributes

Attribute	Selectable Values	Value for 1080p Configuration (Default)	Value for 4K Configuration
General			
Data Type	RAW8, RAW10, RAW12, RAW14, RAW16, RGB565, RGB888, YUV420_8, YUV420_8_CSPS, LEGACY_YUV420_8, YUV420_10, YUV420_10_CSPS, YUV422_8, YUV422_10	RAW10	RAW10
Byte Interface			
RX interface	CSI-2, DSI	CSI-2	CSI-2
Number of RX Lanes	1, 2, 4	4	4
RX Gear	8, 16	8	8
Byte Clock Frequency (MHz)	10–250	46.40625	92.81250
Enable AXI-4 Stream Rx Interface	Enabled, Disabled	Disabled	Disabled
Clock			
Number of Output Pixel Lanes	1, 2, 4	2	2
Camera/Display Control Polarity	Positive, Negative	Positive	Positive
Pixel Clock Frequency (MHz)	10–250	74.25	148.50
Enable AXI-4 Stream Tx Interface	Enabled, Disabled	Disabled	Disabled
Miscellaneous			
Enable Debug Ports	Enabled, Disabled	Disabled	Disabled
Data Settle Cycle	1–27	4	4
FIFO			
Manual Adjust	Enabled, Disabled	Disabled	Disabled
FIFO Implementation	EBR, LUT	EBR	EBR
Word Count	1–65535	2400	4800

3.1.3. Debayer IP

CMOS color image sensors do not capture all the three color components for each pixel, but only one of the three color components for any pixel. Since green is the dominant component that closely captures the luminescence compared to red or blue components, half of the pixels in a sensor capture green component while a fourth of the sensor pixels are used to capture each blue or red component. The process of interpolating and recreating the missing color components which is not captured by the sensor is called Debayering or De-Mosaicing or Color Filter Array (CFA) Interpolation. After Debayering, each pixel is represented by all the three color components. The configuration GUI for the IP is shown in Figure 3.3 and the attributes are described in Table 3.3. For more information, refer to the Debayer IP User Guide (FPGA-IPUG-02203).

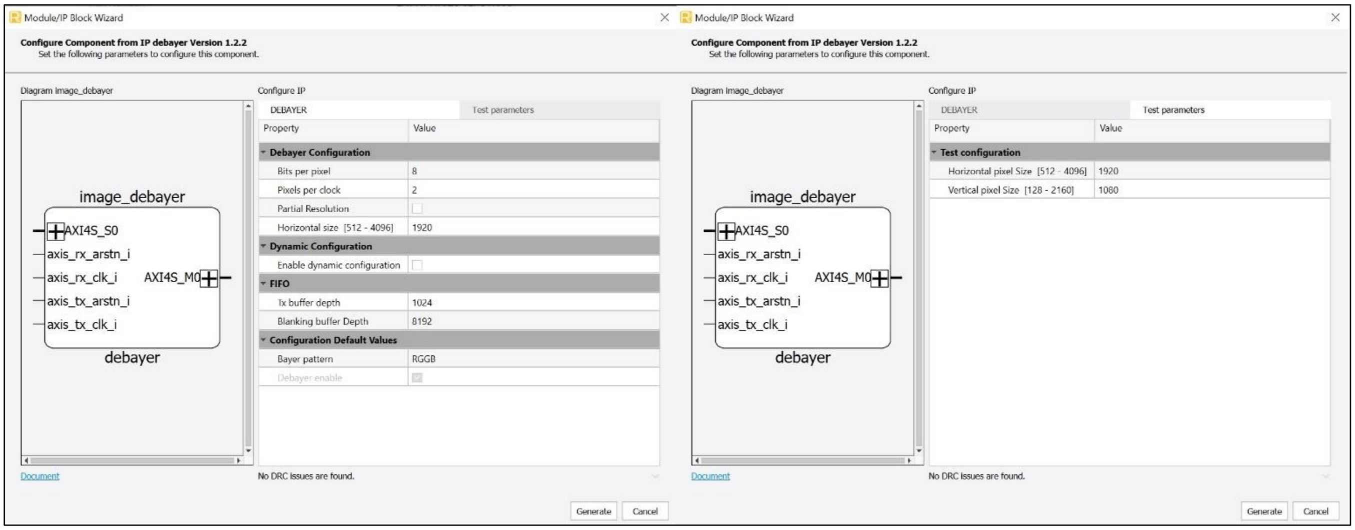

Figure 3.3. Module/IP Block Wizard of Debayer IP

Table 3.3. Debayer IP Attributes

Attribute	Selectable Values	Value for 1080p Configuration (Default)	Value for 4K Configuration
Debayer			
Debayer Configuration			
Bits per Pixel	6, 8, 10, 12, 16	8	8
Pixels per clock	1, 2, 4	2	2
Partial Resolution	Enabled, Disabled	Disabled	Disabled
Horizontal Size	512–4096	1920	3840
Dynamic Configuration			
Enable dynamic configuration	Enabled, Disabled	Disabled	Disabled
FIFO			
Tx buffer depth	128, 256, 512, 1024	1024	1024
Blanking buffer depth	128, 512, 1024, 2048, 4096, 8192, 10240	8192	8192
Configuration Default Values			
Bayer Pattern	RGGB, BGGR, GRBG, GBRG	RGGB	RGGB
Debayer enable	Enabled, Disabled	Enabled	Enabled
Test parameters			
Test configuration			
Horizontal pixel size	512–4096	1920	3840
Vertical pixel size	128–2160	1080	2160

3.1.4. Automatic White Balance IP

The Automatic White Balance (AWB) IP automatically compensates for illumination temperature-based color differences, so that white actually appears white. The operation is performed mostly in Bayer domain, but it can also be done in RGB domain. AWB is a pixel-based operation that uses image statistics. Since pixel correction is scene based, it does not require calibration. White balancing is a two-step process: determination of the nature of illuminant and image correction based on the illuminant. The configuration GUI for the IP is shown in Figure 3.4 and the attributes are described in Table 3.4. For more information, refer to the Automatic White Balance IP User Guide (FPGA-IPUG-02204).

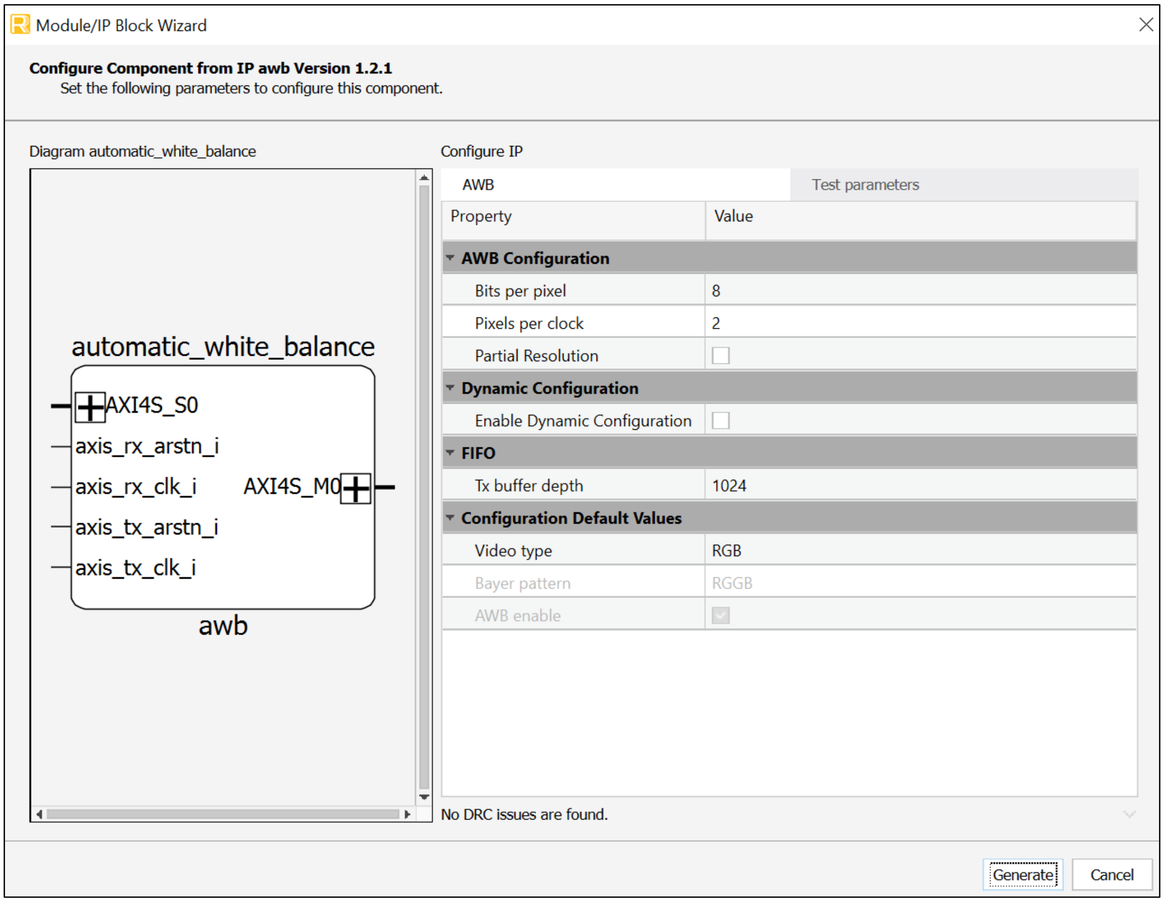

Figure 3.4. Module/IP Block Wizard of Automatic White Balance IP

Table 3.4. Automatic White Balance IP Attributes

Attribute	Selectable Values	Value for 1080p Configuration (Default)	Value for 4K Configuration
Automatic White Balance			
Automatic White Balance Configuration			
Bits per Pixel	6, 8, 10, 12, 16	8	8
Pixels per clock	1, 2, 4	2	2
Partial Resolution	Enabled, Disabled	Disabled	Disabled
Dynamic Configuration			
Enable dynamic configuration	Enabled, Disabled	Disabled	Disabled
FIFO			
Tx buffer depth	128, 256, 512, 1024	1024	1024
Configuration Default Values			
Video Type	RGB, Bayer	RGB	RGB
Bayer Pattern	RGGB, BGGR, GRBG, GBRG	RGGB	RGGB
Automatic White Balance enable	Enabled, Disabled	Enabled	Enabled
Test parameters			
Test configuration			
Horizontal pixel size	512–4096	1920	3840
Vertical pixel size	128–2160	1080	2160

3.1.5. Color Correction Matrix IP

The measured RGB values of the image sensors are different from the true RGB values of the image. This difference is mostly attributable to the characteristics of the optical filter overlay in the sensor. To obtain the correct colors, the pixels need to be mapped from sensor RGB color space to standard RGB color space. This linear mapping of the color components is achieved using a 3x3 matrix, called color correction matrix (CCM). CCM is a pixel-level operation which does not require any line buffers. The operation is done in the RGB domain. The configuration GUI for the IP is shown in Figure 3.5 and the attributes are shown in Table 3.5. For more information, refer to the Color Correction Matrix IP User Guide (FPGA-IPUG-02214).

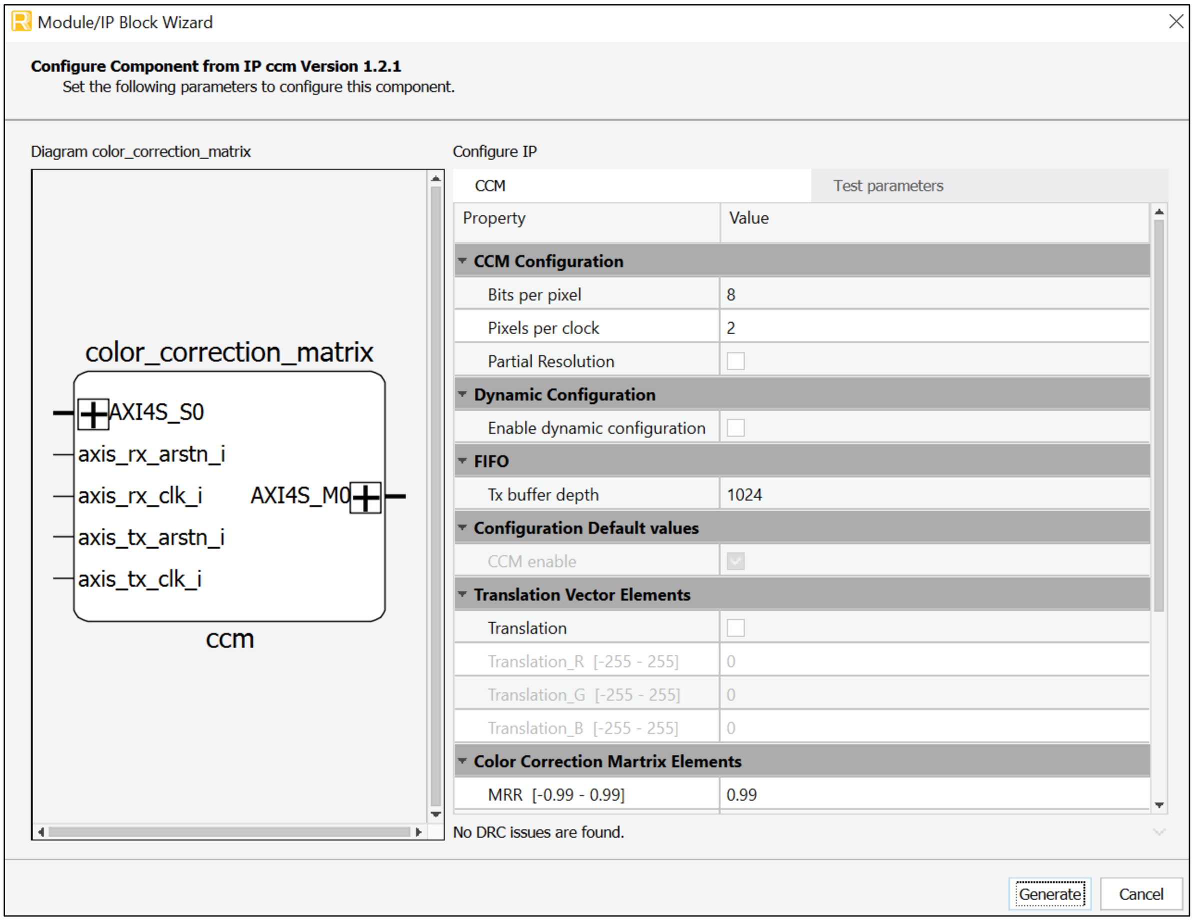

Figure 3.5. Module/IP Block Wizard of Color Correction Matrix IP

Table 3.5. Color Correction Matrix IP Attributes

Attribute	Selectable Values	Value for 1080p Configuration (Default)	Value for 4K Configuration
Color Correction Matrix			
Color Correction Matrix Configuration			
Bits per Pixel	6, 8, 10, 12, 16	8	8
Pixels per clock	1, 2, 4	2	2
Partial Resolution	Enabled, Disabled	Disabled	Disabled
Dynamic Configuration			
Enable dynamic configuration	Enabled, Disabled	Disabled	Disabled
FIFO			
Tx buffer depth	128, 256, 512, 1024	1024	1024

Attribute	Selectable Values	Value for 1080p Configuration (Default)	Value for 4K Configuration
Configuration Default Values			
Color Correction Matrix enable	Enabled, Disabled	Enabled	Enabled
Translation Vector Elements			
Translation	Enabled, Disabled	Disabled	Disabled
Translation_R	-255–255	0	0
Translation_G	-255–255	0	0
Translation_B	-255–255	0	0
Color Correction Matrix Elements			
MRR	-0.99–0.99	0.99	0.99
MRG	-0.99–0.99	0	0
MRB	-0.99–0.99	0	0
MGR	-0.99–0.99	0	0
MGG	-0.99–0.99	0.99	0.99
MGB	-0.99–0.99	0	0
MBR	-0.99–0.99	0	0
MBG	-0.99–0.99	0	0
MBB	-0.99–0.99	0.99	0.99
Test parameters			
Test configuration			
Horizontal pixel size	512–4096	1920	3840
Vertical pixel size	128–2160	1080	2160

3.1.6. Video Frame Buffer IP

Some of the sensors do not have a consistent horizontal or vertical blanking, which may violate timing specifications of the display devices. The Video Frame Buffer IP is inserted to buffer a whole frame into an external DRAM, so the downstream logics can independently generate the required timings for the display devices. The configuration GUI for the IP is shown in Figure 3.6 and the attributes are shown in Table 3.6.

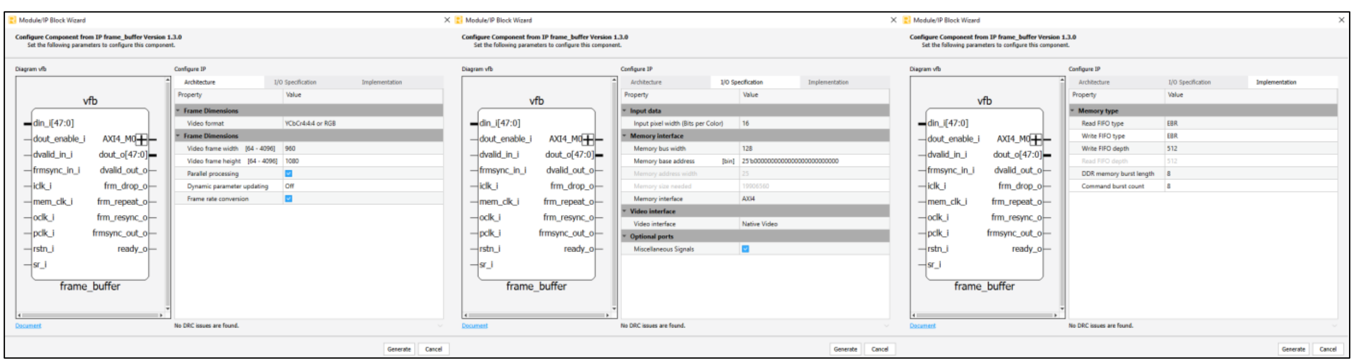

Figure 3.6. Module/IP Block Wizard of Video Frame Buffer IP

Table 3.6. Video Frame Buffer IP Attributes

Attribute	Selectable Values	Value for 1080p Configuration (Default)	Value for 4K Configuration
Architecture			
Frame Dimensions			
Video format	Single Color, YbCbCr4:2:2, YbCbCr4:4:4 or RGB	YbCbCr4:4:4 or RGB	YbCbCr4:4:4 or RGB
Video frame width[1]	64–4096	960	1920
Video frame height	64–4096	1080	2160
Parallel processing	Checked, Unchecked	Checked	Checked
Dynamic parameter updating	On, Off	Off	Off
Frame rate conversion	Checked, Unchecked	Checked	Checked
I/O Specification			
Input Data			
Input pixel width (Bits per Color)[1]	8, 10, 12, 16	16	16
Memory Interface			
Memory bus width	64, 128	128	128
Memory base address	0 to ($2^{25} – 1$)	0x0	0x0
Memory interface	AXI4, Native Interface	AXI4	AXI4
Video Interface			
Video Interface	Native Video, Unified Video Interface	Native Video	Native Video
Optional Ports			
Miscellenaneous Signals	Checked, Unchecked	Checked	Checked
Implementation			
Memory Type			
Read FIFO type	EBR, Distributed	EBR	EBR
Write FIFO type	EBR, Distributed	EBR	EBR
Write FIFO depth	64, 128, 256, 512	512	512
DDR memory burst length	2, 4, 8	8	8
Command burst count	1, 2, 4, 8	8	8

Note:

1. The input and output of the IP module are 48-bit wide (2 pixels per clock of 24-bit RGB888), thus the Video Frame width for the IP is half of the intended resolution. The IP does not officially support 2 pixels per clock format; thus, we are using 16-bit pixel width (bits per color) as a workaround.

3.1.7. LPDDR4 Memory Controller IP

To handle the communication between the Video Frame Buffer IP and the external LPDDR4, LPDDR4 Memory Controller IP is instantiated. The configuration for both 1080p and 4K are equivalent. A dedicated 100 MHz on-board clock is directly supplied to the IP. The configuration GUI for the IP is shown in Figure 3.7 and the attributes are shown in Table 3.7.

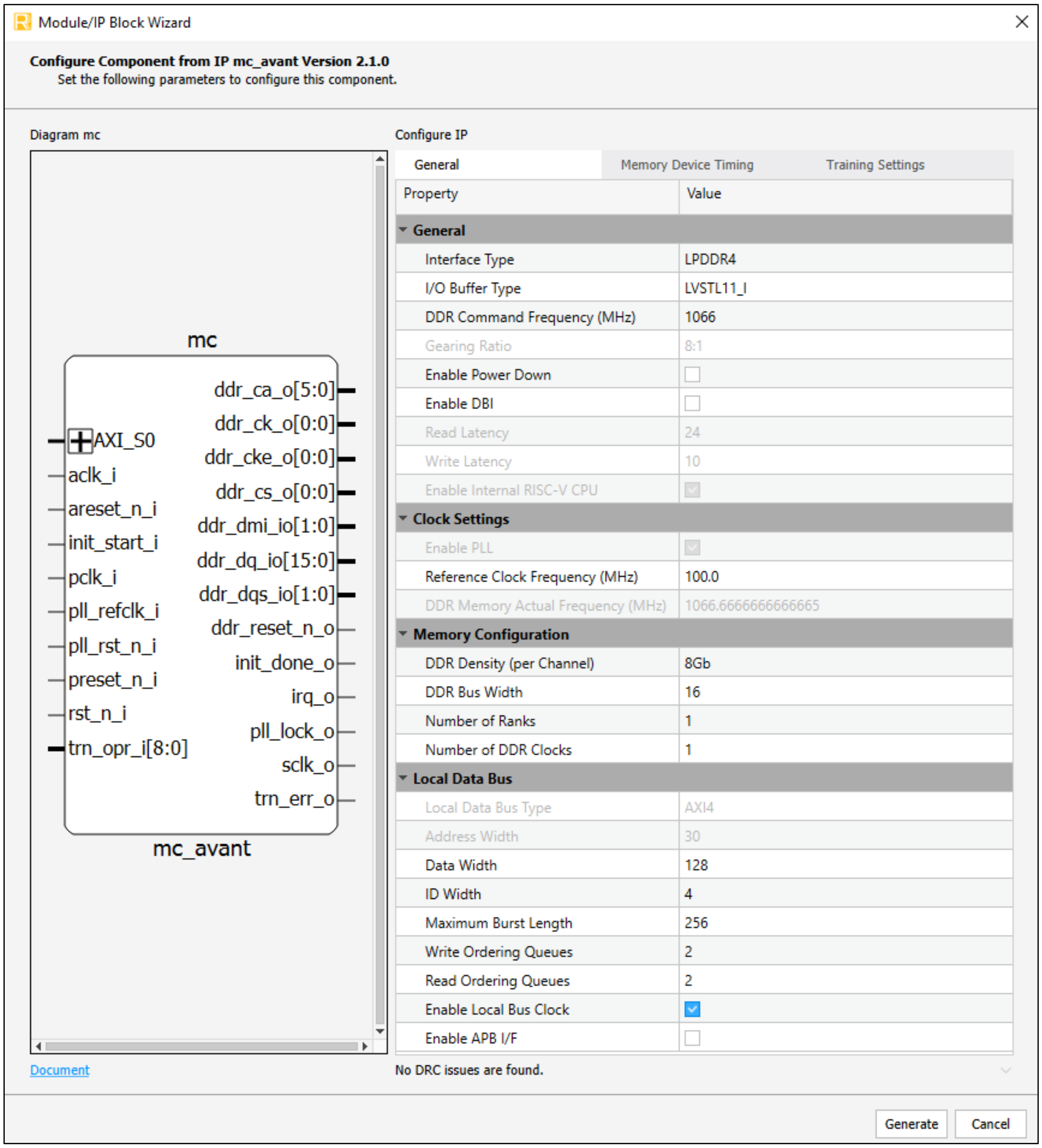

Figure 3.7. Module/IP Block Wizard of LPDDR4 Memory Controller IP

Table 3.7. LPDDR4 Memory Controller IP Attributes[1]

Attribute	Selectable Values	Reference Design Fixed Value
General		
General		
Interface Type	LPDDR4, DDR4	LPDDR4
I/O Buffer Type	LVSTL11_I, LVSTL11_II	LVSTL11_I
DDR Command Frequency (MHz)	350 – 1200	1066
Enable Power Down	Checked, Unchecked	Unchecked
Enable DBI	Checked, Unchecked	Unchecked
Clock Settings		
Reference Clock Frequency (MHz)	25, 50, 100	100

Attribute	Selectable Values	Reference Design Fixed Value
Memory Configuration		
DDR Density (per Channel)	2, 4, 8, 16	8
DDR Bus Width	16, 32, 64	16
Number of Ranks	1, 2	1
Number of DDR Clocks	1, 2	1
Local Data Bus		
Data Width[2]	32, 64, 128	128
ID Width	2 – 8	4
Maximum Burst Length	64, 128, 256	256
Write Ordering Queues	1 – 4	2
Read Ordering Queues	1 – 4	2
Enable Local Bus Clock	Checked, Unchecked	Checked
Enable APB I/F	Checked, Unchecked	Unchecked

Notes:

1. The other configurations such as Memory Device Timing and Training Settings are being left as default.
2. *Data Width* must match *Memory bus width* parameter of the Video Frame Buffer IP.

3.1.8. System Clocks PLL IP

This PLL IP is used to generate clocks for the image sensor, I2C modules, and sync clock input for D-PHY Receiver. It takes the 27 MHz clock input from the board. Configuration GUI for this IP is shown in Figure 3.8 and the attributes in Table 3.8. For more information, refer to the PLL Module User Guide (FPGA-IPUG-02220). The clock assignments in the design is as follows:

- CLKOP – 27 MHz System Clock for the I2C programming and sensor reference clock.
- CLKOS – Pixel clock for the pixel data in 2 pixels per clock format.
- CLKOS2 – 90 MHz PCLK for the LPDDR4 Memory Controller IP.
- CLKOS3 – HDMI clock for the pixel data in 1 pixel per clock format.
- CLKOS4 – 66 MHz clock for the D-PHY Rx IP sync clock.

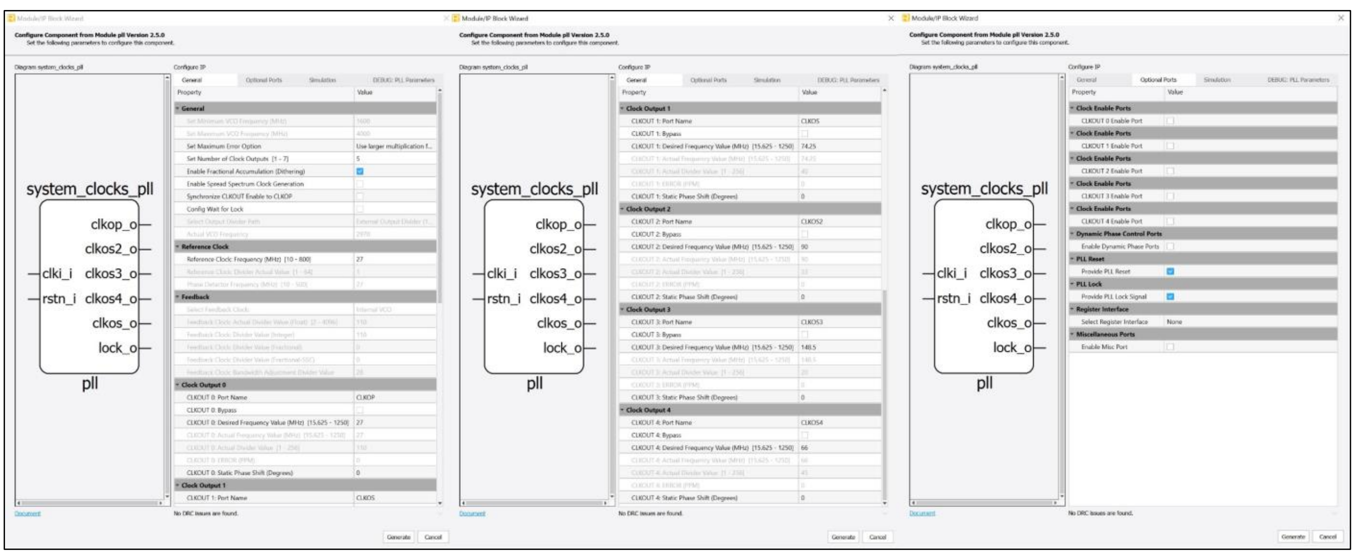

Figure 3.8. Module/IP Block Wizard of System Clocks PLL IP

Table 3.8. System Clocks PLL IP Attributes

Attribute	Selectable Values	Value for 1080p Configuration (Default)	Value for 4k Configuration
General			
General			
Set Maximum Error Option	Use large multiplication factor to reduce error, Use reasonable tolerance (NF ≤ 128), Find solution with lowest error, Use specified tolerance value	Find solution with lowest error	Find solution with lowest error
Set Number of Clock Outputs	1–7	5	5
Enable Fractional Accumulation (Dithering)	Enabled, Disabled	Enabled	Enabled
Enable Spread Spectrum Clock Generation	Enabled, Disabled	Disabled	Disabled
Synchronize CLKOUT Enable to CLKOP	Enabled, Disabled	Disabled	Disabled
Reference Clock			
Reference Clock Frequency[1]	10–800	27	27
Clock Output 0			
CLKOUT 0: Port Name	CLKOP, CLKOPHY, CLKOS, CLKOS2, CLKOS3, CLKOS4, CLKOS5	CLKOP	CLKOP
CLKOUT 0: Bypass	Enabled, Disabled	Disabled	Disabled
CLKOUT 0: Desired Frequency Value[2]	9.765625–1250	27	27
CLKOUT 0: Static Phase Shift (Degrees)	0, 45, 90, 135, 180, 225, 270, 315	0	0
Clock Output 1			
CLKOUT 1: Port Name	CLKOP, CLKOPHY, CLKOS, CLKOS2, CLKOS3, CLKOS4, CLKOS5	CLKOS	CLKOS
CLKOUT 1: Bypass	Enabled, Disabled	Disabled	Disabled
CLKOUT 1: Desired Frequency Value[3]	9.765625–1250	74.25	148.5
CLKOUT 1: Static Phase Shift (Degrees)	0, 45, 90, 135, 180, 225, 270, 315	0	0
Clock Output 2			
CLKOUT 2: Port Name	CLKOP, CLKOPHY, CLKOS, CLKOS2, CLKOS3, CLKOS4, CLKOS5	CLKOS2	CLKOS2
CLKOUT 2: Bypass	Enabled, Disabled	Disabled	Disabled
CLKOUT 2: Desired Frequency Value[4]	9.765625–1250	90	90
CLKOUT 2: Static Phase Shift (Degrees)	0, 45, 90, 135, 180, 225, 270, 315	0	0
Clock Output 3			
CLKOUT 3: Port Name	CLKOP, CLKOPHY, CLKOS, CLKOS2, CLKOS3, CLKOS4, CLKOS5	CLKOS3	CLKOS3
CLKOUT 3: Bypass	Enabled, Disabled	Disabled	Disabled
CLKOUT 3: Desired Frequency Value[5]	9.765625–1250	148.5	297
CLKOUT 3: Static Phase Shift (Degrees)	0, 45, 90, 135, 180, 225, 270, 315	0	0

Attribute	Selectable Values	Value for 1080p Configuration (Default)	Value for 4k Configuration
Clock Output 4			
CLKOUT 4: Port Name	CLKOP, CLKOPHY, CLKOS, CLKOS2, CLKOS3, CLKOS4, CLKOS5	CLKOS4	CLKOS4
CLKOUT 4: Bypass	Enabled, Disabled	Disabled	Disabled
CLKOUT 4: Desired Frequency Value[6]	9.765625–1250	66	66
CLKOUT 4: Static Phase Shift (Degrees)	0, 45, 90, 135, 180, 225, 270, 315	0	0
Optional Ports			
Clock Enable Ports			
CLKOUT 0 Enable Port	Enabled, Disabled	Disabled	Disabled
CLKOUT 1 Enable Port	Enabled, Disabled	Disabled	Disabled
CLKOUT 2Enable Port	Enabled, Disabled	Disabled	Disabled
CLKOUT 3 Enable Port	Enabled, Disabled	Disabled	Disabled
CLKOUT 4 Enable Port	Enabled, Disabled	Disabled	Disabled
Dynamic Phase Control Ports			
Enable Dynamic Phase Ports	Enabled, Disabled	Disabled	Disabled
PLL Reset or Lock			
Provide PLL Reset	Enabled, Disabled	Enabled	Enabled
Provide PLL Lock Signal	Enabled, Disabled	Enabled	Enabled
Miscellaneous Port			
Enable Misc Port	Enabled, Disabled	Disabled	Disabled
Simulation			
Simulation Type	RTL_SIM, GATE_SIM	RTL_SIM	RTL_SIM

Notes:

1. This 27 MHz clock input comes from the Lattice Modular FMC adapter card.
2. This clock output is used by the image sensor and I2C modules.
3. This clock is used in pixel domain, including the ISP pipelines.
4. This clock is used as the pclk for the memory controller IP module.
5. This clock is HDMI clock, which will be forwarded to the HDMI FMC card too.
6. This clock is used as a sync clock for the RX D-PHY IP module.

3.1.9. ROM IP for Image Sensor I2C ROM 2K

This ROM IP stores the I2C commands used to configure the image sensor for 1080p resolution in the required sequence. The configuration GUI for the ROM IP is shown in Figure 3.9 and the attributes are shown in Table 3.9. For more information, refer to the EBR Memory Modules User Guide (FPGA-IPUG-02190).

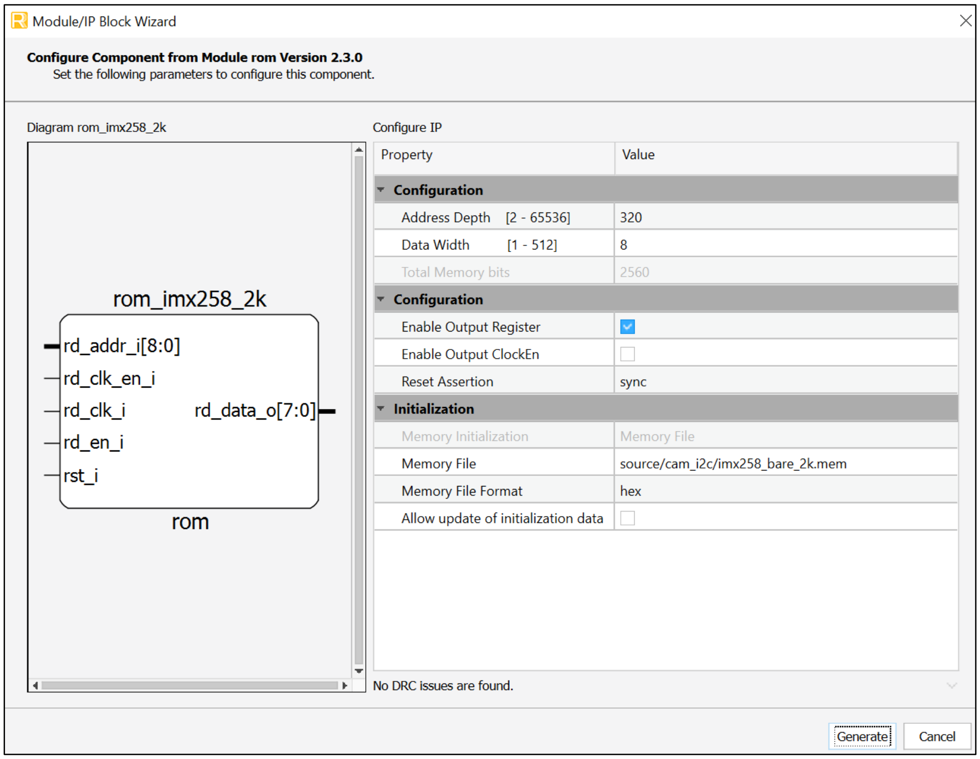

Figure 3.9. Module/IP Block Wizard of Image Sensor I2C ROM 2K IP

Table 3.9. Image Sensor I2C ROM 2K IP Attributes

Attribute	Selectable Values	Default Value
Address Depth	2–65536	320
Data Width	1–512	8
Enable Output Register	Enabled, Disabled	Enabled
Enable Output ClockEn	Enabled, Disabled	Disabled
Reset Assertion	async, sync	sync
Memory File	File Path	source/cam_i2c/imx258_bare_2k.mem
Memory File Format	hex, binary	hex
Allow update of initialization data	Enabled, Disabled	Disabled

3.1.10. ROM IP for Image Sensor I2C ROM 4K

This ROM IP stores the I2C commands used to configure the image sensor for 4K resolution in the required sequence. The configuration GUI for the IP is shown in Figure 3.10 and the attributes are described in Table 3.10. For more information, refer to the EBR Memory Modules User Guide (FPGA-IPUG-02190).

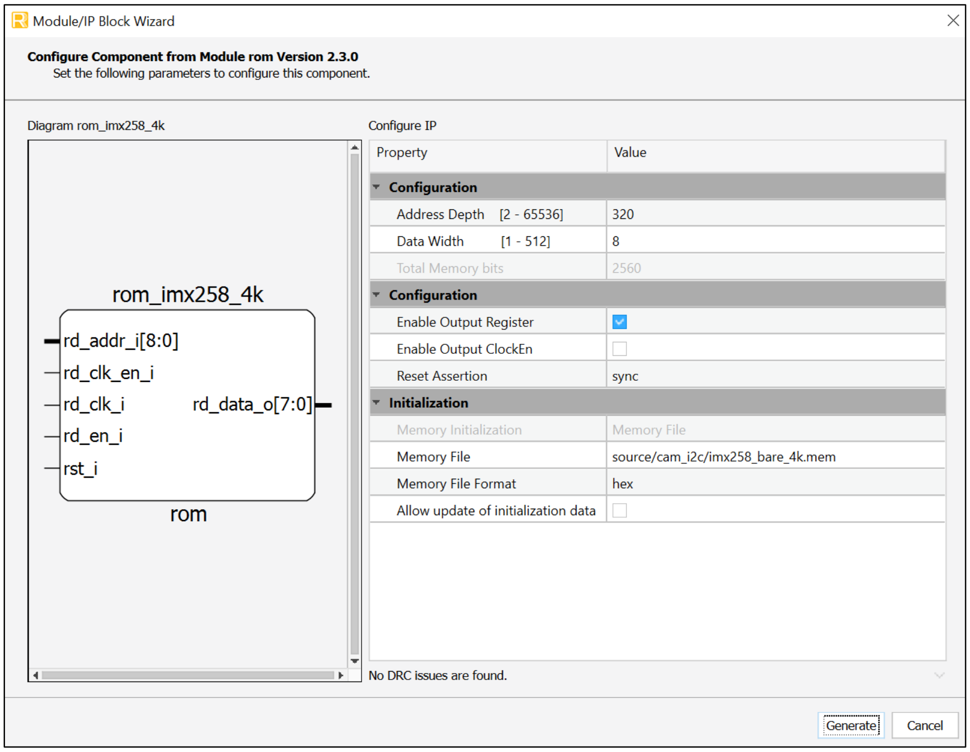

Figure 3.10. Module/IP Block Wizard of Image Sensor I2C ROM 4K IP

Table 3.10. Image Sensor I2C ROM 4K IP Attributes

Attribute	Selectable Values	Default Value
Address Depth	2–65536	320
Data Width	1–512	8
Enable Output Register	Enabled, Disabled	Enabled
Enable Output ClockEn	Enabled, Disabled	Disabled
Reset Assertion	async, sync	sync
Memory File	File Path	source/cam_i2c/imx258_bare_4k.mem
Memory File Format	hex, binary	hex
Allow update of initialization data	Enabled, Disabled	Disabled

3.1.11. ROM IP for HDMI I2C ROM 2K

This ROM IP stores the I2C commands used to configure the HDMI for 1080p resolution in the required sequence. The configuration GUI for the IP is shown in Figure 3.11 and the attributes in Table 3.11. For more information, refer to the EBR Memory Modules User Guide (FPGA-IPUG-02190).

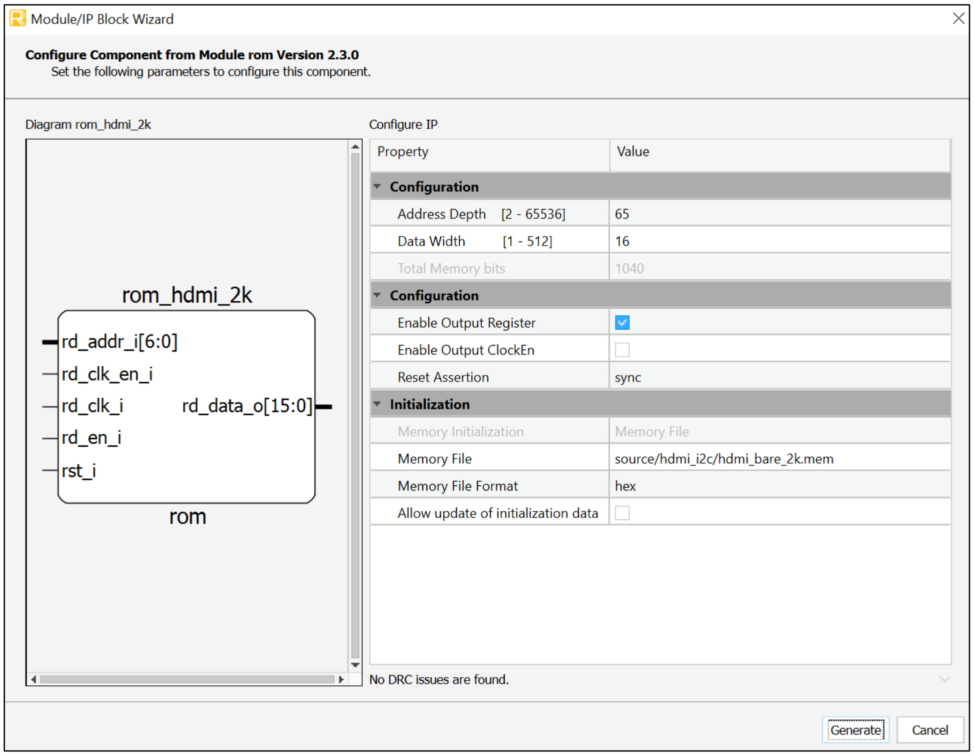

Figure 3.11. Module/IP Block Wizard of HDMI I2C ROM 2K IP

Table 3.11. HDMI I2C ROM 2K IP Attributes

Attribute	Selectable Values	Default Value
Address Depth	2–65536	65
Data Width	1–512	16
Enable Output Register	Enabled, Disabled	Enabled
Enable Output ClockEn	Enabled, Disabled	Disabled
Reset Assertion	async, sync	sync
Memory File	File Path	source/hdmi_i2c/hdmi_bare_2k.mem
Memory File Format	hex, binary	hex
Allow update of initialization data	Enabled, Disabled	Disabled

3.1.12. ROM IP for HDMI I2C ROM 4K

This ROM IP stores the I2C commands used to configure the HDMI for 4K resolution in the required sequence. The configuration GUI for the IP is shown in Figure 3.12 and the attributes are described in Table 3.12. For more information, refer to the EBR Memory Modules User Guide (FPGA-IPUG-02190).

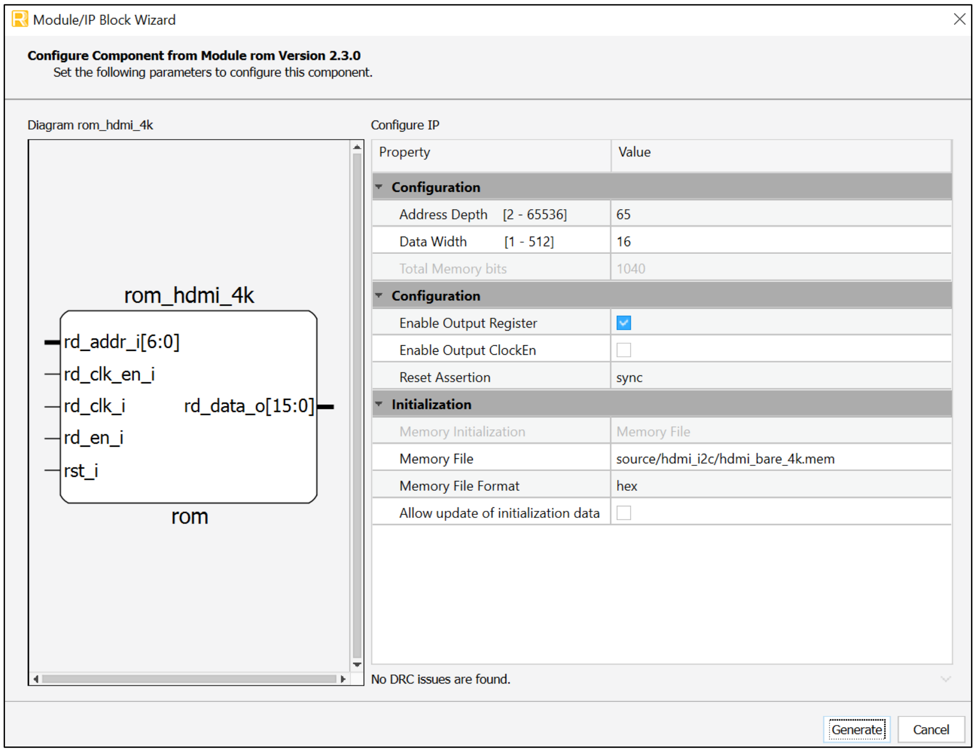

Figure 3.12. Module/IP Block Wizard of HDMI I2C ROM 4K IP

Table 3.12. HDMI I2C ROM 4K IP Attributes

Attribute	Selectable Values	Default Value
Address Depth	2–65536	65
Data Width	1–512	16
Enable Output Register	Enabled, Disabled	Enabled
Enable Output ClockEn	Enabled, Disabled	Disabled
Reset Assertion	async, sync	sync
Memory File	File Path	source/hdmi_i2c/hdmi_bare_4k.mem
Memory File Format	hex, binary	hex
Allow update of initialization data	Enabled, Disabled	Disabled

3.2. Signal Descriptions

This section describes the signals for the CSI-2 to HDMI core design and the Demonstration design.

3.2.1. CSI-2 to HDMI Core Design

Table 3.13 shows the input/output interface signals for the core design (MIPI_RX_HDMI_TOP module).

Table 3.13. CSI-2 to HDMI Core Design Signals

Port Name	Type	Width	Description
reset_n_i	Input	1	Active low asynchronous system reset.
sync_clk_i	Input	1	Low speed or oscillator clock. Operating frequency of the components interfaced with the fabric.
sync_rst_i	Input	1	Active high, synchronized reset.
reset_byte_fr_n_i	Input	1	Low asserted reset for the nets in the DPHY Rx byte clock domain.

Port Name	Type	Width	Description
reset_pixel_n_i	Input	1	Low asserted reset for the nets in the Byte-to-Pixel Converter IP pixel clock domain.
clk_pixel_i	Input	1	Pixel clock at which Byte-to-Pixel Converter IP produces output pixels.
clk_pixel_debayer_i	Input	1	Pixel clock for the ISP such as the Debayer IP. Set to the same clock as clk_pixel_i.
clk_hdmi_i	Input	1	Pixel clock used to generate HDMI video timings.
reset_hdmi_n_i	Input	1	Low asserted reset for the nets in the HDMI clock domain.
clk_mcddr_ref_i	Input	1	100 MHz reference clock used by the LPDDR4 Memory Controller IP.
clk_p_io	Input/Output	1	MIPI D-PHY positive clock lane.
clk_n_io	Input/Output	1	MIPI D-PHY negative clock lane.
d_p_io	Input/Output	4	MIPI D-PHY positive data lane.
d_n_io	Input/Output	4	MIPI D-PHY negative data lane.
ddr_dq_io	Input/Output	16	LPDDR4 DQ signal.
ddr_dqs_io	Input/Output	2	LPDDR4 DQS signal.
ddr_dmi_io	Input/Output	2	LPDDR4 DMI signal.
clk_byte_hs_o	Output	1	MIPI D-PHY receiver output byte clock.
data_o	Output	24	HDMI output data payload.
vsync_o	Output	1	HDMI output vertical sync signal.
hsync_o	Output	1	HDMI output horizontal sync signal.
de_o	Output	1	HDMI output data enable.
ddr_ck_o	Output	2	LPDDR4 CK signal.
ddr_cke_o	Output	1	LPDDR4 CKE signal.
ddr_cs_o	Output	1	LPDDR4 CS signal.
ddr_ca_o	Output	6	LPDDR4 CA signal.
ddr_odt_ca_o	Output	1	LPDDR4 CA signal.
ddr_reset_n_o	Output	1	Memory reset signal.
pclk_i	Input	1	PCLK for the LPDDR4 Memory Controller IP.
prstn_i	Input	1	Low asserted reset for the nets in the PCLK clock domain.
mc_train_err_o	Output	1	Debug signal. Asserted if there is an error during the LPDDR4 training. Sticky until the next reset.
mc_init_done_o	Output	1	Debug signal. Asserted when the LPDDR4 initialization and training are done. Sticky until the next reset.
b2p_deb_line_err_o	Output	1	Debug signal. Asserted when the horizontal blanking from the sensor is not sufficient causing overlap between lines. Sticky until the next reset.
b2p_deb_frame_err_o	Output	1	Debug signal. Asserted when the vertical blanking from the sensor is not sufficient causing overlap between frames. Sticky until the next reset.

3.2.2. Demonstration Design

Table 3.14 shows the input/output interface signals for the demonstration design (FPGA_TOP module).

Table 3.14. Demonstration Design Signals

Port Name	Type	Width	Description
reset_n_i	Input	1	Active low asynchronous system reset; connected to SW1 push button.
clk_i	Input	1	27 MHz input clock for the system clock PLL.
mcddr_refclk_i	Input	1	100 MHz input clock for the LPDDR4 Memory Controller IP.
cam_scl_io	Input/Output	1	Image sensor I2C configuration serial clock lane.
cam_sda_io	Input/Output	1	Image sensor I2C configuration serial data lane.

Port Name	Type	Width	Description
hdmi_scl_io	Input/Output	1	HDMI I2C configuration serial clock lane.
hdmi_sda_io	Input/Output	1	HDMI I2C configuration serial data lane.
clk_p_io	Input/Output	1	MIPI D-PHY positive clock lane.
clk_n_io	Input/Output	1	MIPI D-PHY negative clock lane.
d_p_io	Input/Output	4	MIPI D-PHY positive data lane.
d_n_io	Input/Output	4	MIPI D-PHY negative data lane.
ddr_dq_io	Input/Output	16	LPDDR4 DQ signal.
ddr_dqs_io	Input/Output	2	LPDDR4 DQS signal.
ddr_dmi_io	Input/Output	2	LPDDR4 DMI signal.
data_o	Output	24	HDMI output data payload.
vsync_o	Output	1	HDMI output vertical sync signal.
hsync_o	Output	1	HDMI output horizontal sync signal.
de_o	Output	1	HDMI output data enable.
hdmi_clk_o	Output	1	Output clock sent to the HDMI FMC board.
cam_clk_o	Output	1	Output clock sent to the Modular FMC adapter.
hdmi_rst_o	Output	1	Output reset sent to the HDMI FMC board.
sensor_reset_n_o	Output	1	Output reset sent to the Modular FMC adapter.
zeros_o	Output	12	Stream of zeros sent to the HDMI along with data.
ddr_ck_o	Output	2	LPDDR4 CK signal.
ddr_cke_o	Output	1	LPDDR4 CKE signal.
ddr_cs_o	Output	1	LPDDR4 CS signal.
ddr_ca_o	Output	6	LPDDR4 CA signal.
ddr_odt_ca_o	Output	1	LPDDR4 CA signal.
ddr_reset_n_o	Output	1	LPDDR4 Memory reset signal.
dbg_led_o	Output	8	Bit 0 – Inversion of the camera I2C programming done signal. Bit 1 – Inversion of the HDMI card I2C programming done signal. Bit 2 – Inversion of the LPDDR4 initialization done signal. Bit 3 – Inversion of the LPDDR4 training error signal. Bit 4 – Inversion of the sensor line error signal. Bit 5 – Inversion of the sensor frame error signal. Bit 6 – Reserved. It is fixed to 1'b1. Bit 7 – Reserved. It is fixed to 1'b1.

4. Opening and Running the Project

This section provides information on how to open the project, run the simulation, and generate the bit file of CSI-2 to HDMI Reference Design using the Lattice Radiant software. For more details on the Lattice Radiant software, refer to the Lattice Radiant Software user guide.

4.1. Opening the Project

This section describes the procedure for running the CSI-2 to HDMI RD project in the Lattice Radiant software.

1. Open the existing project file, *MIPI_HDMI_RD.rdf* by clicking on the *Open Project* icon present on the home screen of the Radiant software as shown in Figure 4.1.

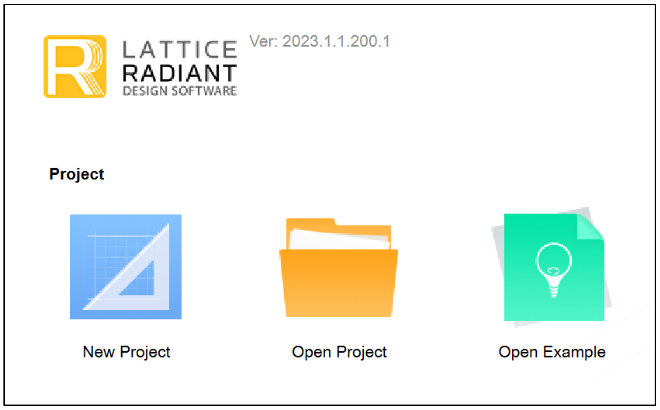

Figure 4.1. Open Project Icon

2. Open the radiant project (.rdf file) in the project directory as shown in Figure 4.2.

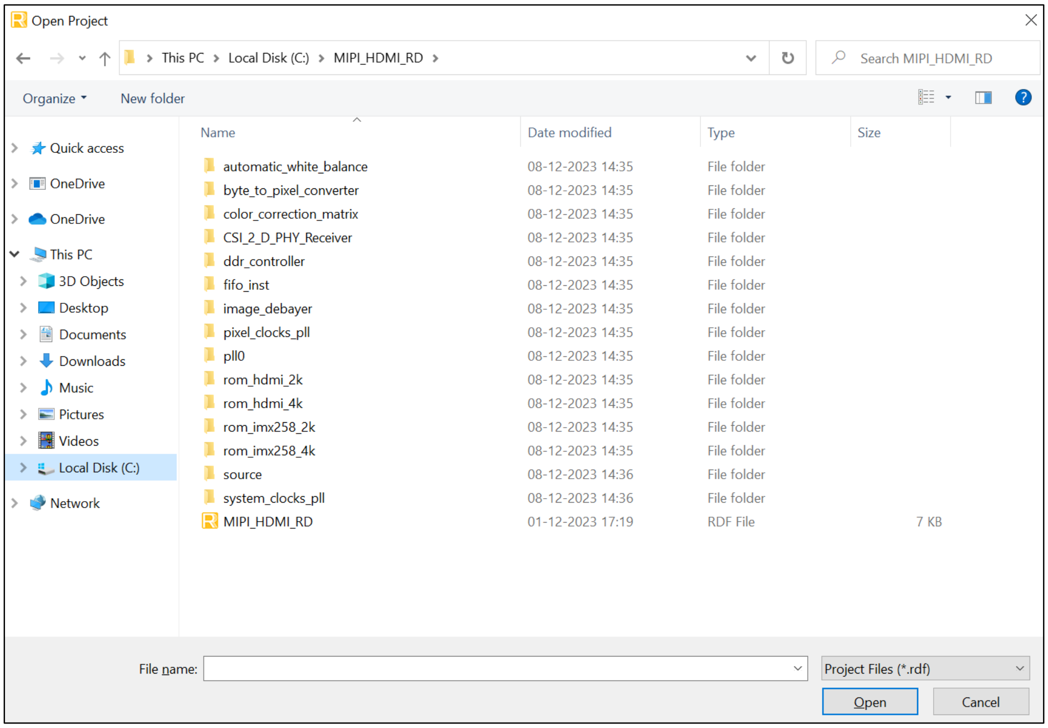

Figure 4.2. Open .rdf Project File

3. In the Tcl console, use the scripts in the *scripts* folder to automatically generate the IPs to the respective configuration. Below is the example of running the script. You may opt out this step and manually regenerate the IPs by doing step 6 and onwards.

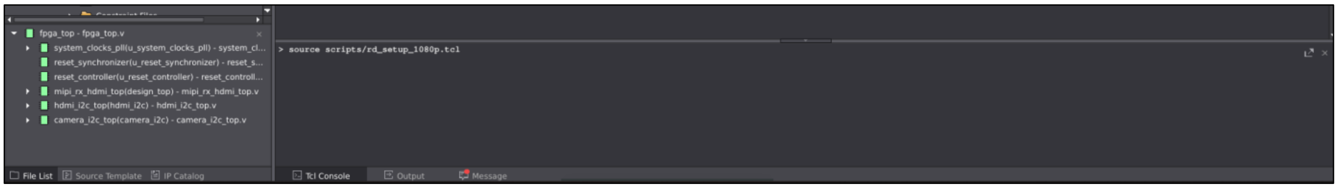

Figure 4.3. Running Tcl Script to Automatically Configure the IP

4. For the manual IP generations, make sure the latest version of the IPs used in the project (except for the Memory Controller IP) are installed locally in your system. If any IP is not installed or has an older version, install the correct version from the server. Then, regenerate all the IPs for the specific configuration as described in the IP Descriptions section. As an example, the regeneration procedure for the CSI-2 D-PHY Receiver IP is given in steps 4 and 5.
5. Double click on the *CSI_2_DPHY_Receiver.ipx* file present in the *File List* section of the Radiant software to configure the DPHY_RX IP as shown in Figure 4.4.

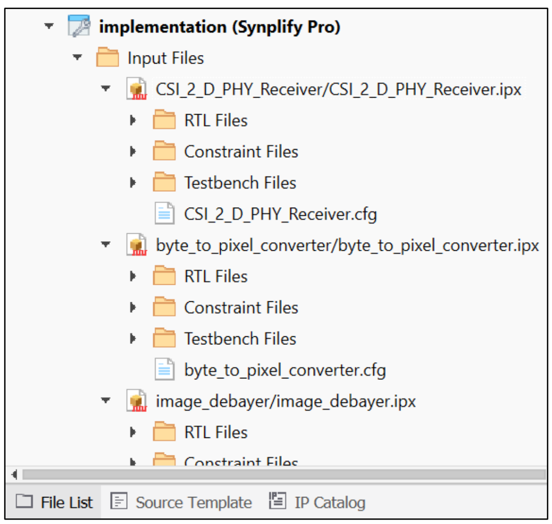

Figure 4.4. Open CSI_2_DPHY_Receiver.ipx File

6. Select the desired parameters from the opened DPHY_RX IP GUI as shown in Figure 4.5.

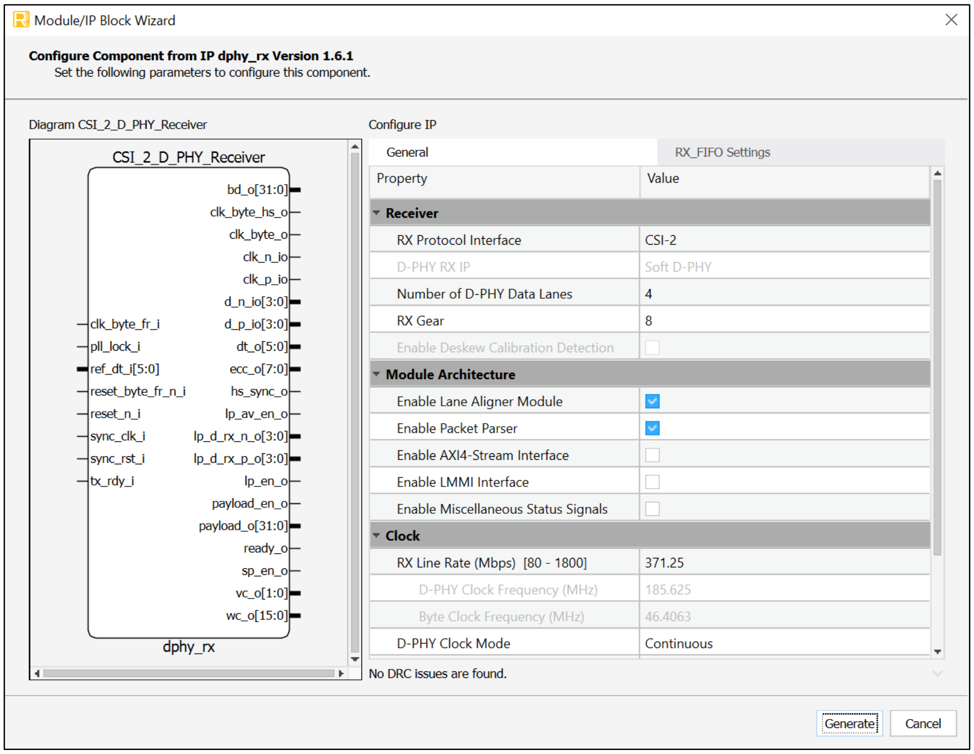

Figure 4.5. Module/IP Block Wizard of dphy_rx IP

7. Configure all the remaining IPs described in the IP Descriptions section in a similar manner.

4.2. Functional Simulation

After opening the project and regenerating the IPs, the testbench needs to be set up before running functional simulation of the design. This is described in the following subsections.

4.2.1. Testbench Setup for Simulation

1. Open the *tb_top.v* file present in the *File List* section in Radiant software as shown in Figure 4.6.

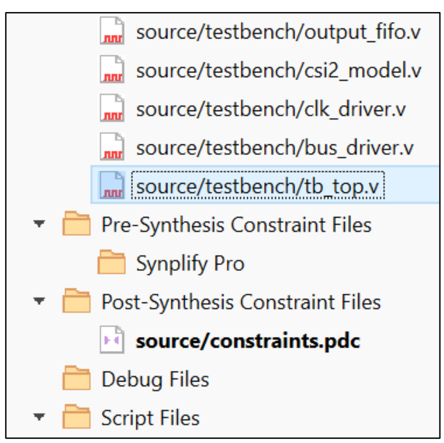

Figure 4.6. Open tb_top.v File

2. Set the parameters defined at the top of the *tb_top* (shown in Figure 4.7) as per the desired configuration and save the file.

```
//User editable parameters
parameter CUSTOM_INPUT = 0;
parameter DEBUG = "ON";
```

Figure 4.7. Set Parameters at the Top of the tb_top

The *DEBUG* parameter when set to "*ON*", prints the output of intermediate IPs into files and displays the data generated by the traffic generator for debugging. If *CUSTOM_INPUT* parameter is set to *0*, the simulation will use the default image Bayerized data present in the directory *image_debayer/testbench/bayer_data.txt* as the input. If *CUSTOM_INPUT* parameter is set to *1* as shown in Table 4.1, the simulation will run for custom input images. Any other setting of *CUSTOM_INPUT* parameters is not allowed.

Table 4.1. TB_TOP Module Parameters

Parameter	Default Value	Description
DEBUG	ON	Enables or disables testbench debug mode.
CUSTOM_INPUT	0	Selects the input mode for simulation.

Things to consider before running functional simulation:

- The simulation can run out of the box for both 4K and 1080p resolutions without changing any testbench parameters using default input image. If *CUSTOM_INPUT* is set to *1*, custom images need to be Bayerized first before they can be passed on to the testbench as described in the section. If the simulation is set up to use custom images as input, the testbench and python script automatically select the appropriate files to generate their outputs.
- Make sure the changes in gear, number of serial data lanes, data type (bits per pixel), PPC, frame resolution, operating frequencies and data rates are reflected in both the *tb_top.v* as well as the GUIs of respective IPs.
- Since the Video Frame Buffer IP buffers the whole frame prior to read, set *NUM_FRAMES* to *2*.

4.2.2. Simulation Setup for Custom Input Images

To input custom image to the testbench, the image must be converted to Bayer format, and the *CUSTOM_INPUT* parameter must be set to *1*. The following procedure describes the process of Bayerizing the custom image. The custom image should be in PNG format.

1. The custom image should be copied to *source/testbench/test_vectors/* directory and it should be renamed to *image.png*.

2. The renamed custom image is converted into Bayerized text file using the *png_to_bayer.py* script present in the folder *source/testbench/test_vectors/png_to_bayer.py*. To run the python script, open the command prompt in the *source/testbench/test_vectors/* directory and enter *python png_to_bayer.py*. After the script is executed, an output text file named *image_bayer_data.txt* is generated in the *test_vectors* folder and *Image bayerized successfully!* message is printed on the screen as shown in Figure 4.8.

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.3693]
(c) Microsoft Corporation. All rights reserved.

C:\MIPI_HDMI_RD\source\testbench\test_vectors>python png_to_bayer.py
Image bayerized successfully!

C:\MIPI_HDMI_RD\source\testbench\test_vectors>
```

Figure 4.8. Generating Bayer Data for Custom Image

Note: Custom image inputs should only be used for visual equivalency comparison, not for an accurate byte-by-byte comparison of input and output data.

4.2.3. Running Functional Simulation

Functional simulation can be performed after the testbench is setup and IPs have been regenerated. Perform the following steps to run the functional simulation.

1. Click the  button located on the *Toolbar* to initiate the *Simulation Wizard* shown in Figure 4.9.

Figure 4.9. Simulation Button Icon

2. Enter the project name in the simulation wizard and click on the *Next* button as shown in Figure 4.10.

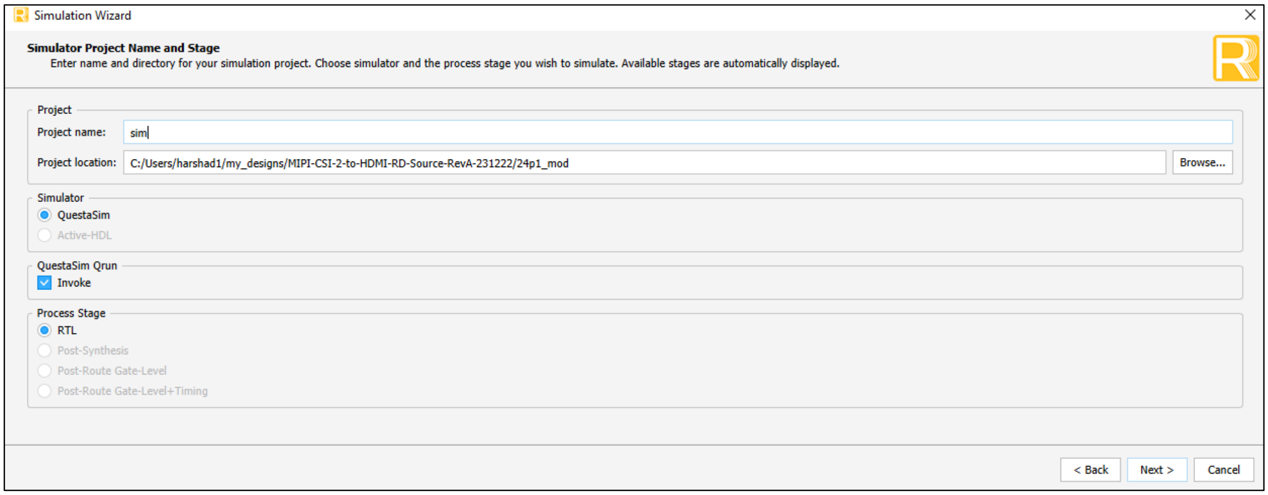

Figure 4.10. Simulation Wizard

3. Note there are errors in the *Parse HDL files for simulation* interface. You can ignore the errors shown in Figure 4.11.

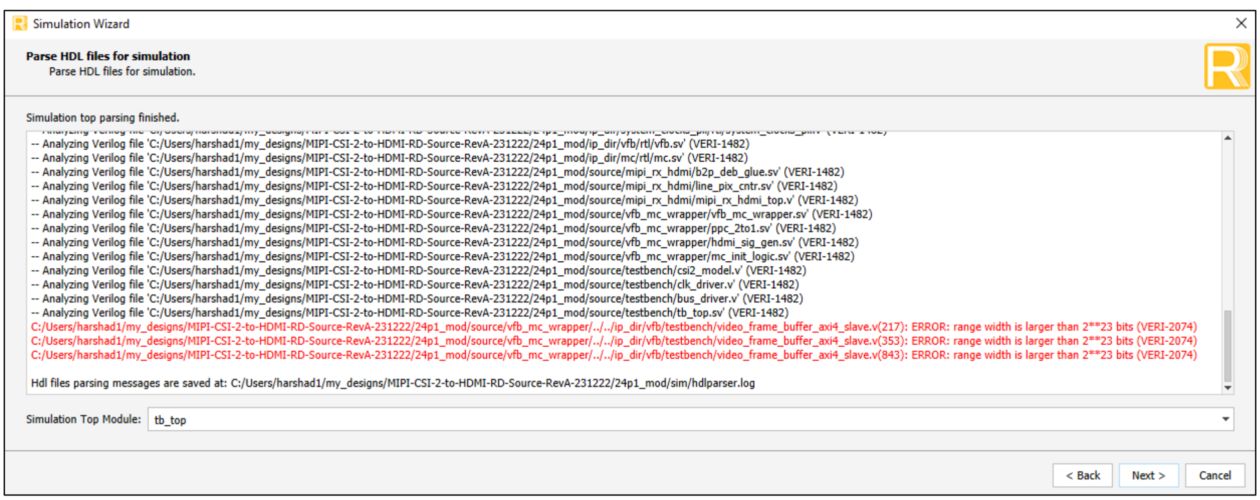

Figure 4.11. Parse HDL Files for Simulation Interface

4. You can change the simulation time to ***0 ns*** to run all the simulations as shown in Figure 4.12. Click the *Finish* button to run the simulation.

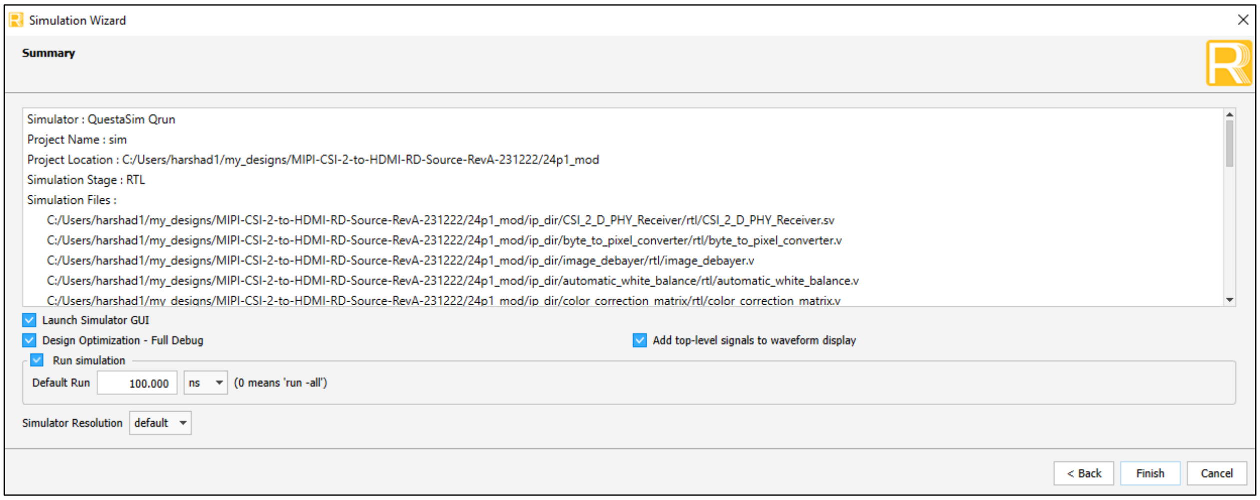

Figure 4.12. Run Simulation Value of 0 to Run All Simulations

5. The simulation waveform is opened as shown in Figure 4.13.

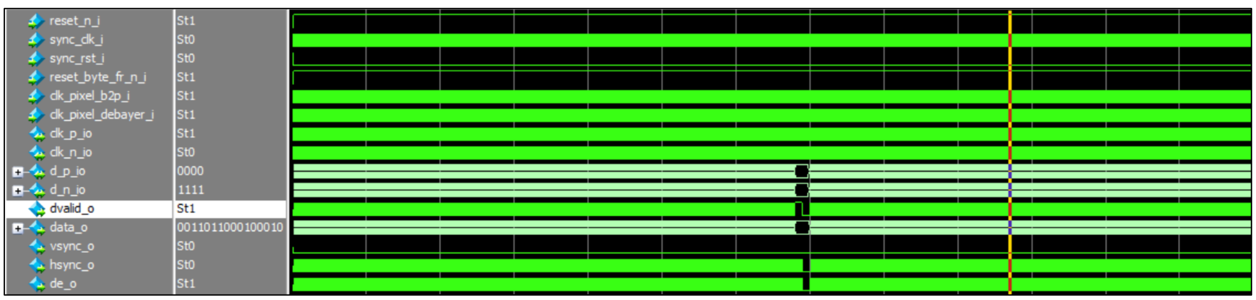

Figure 4.13. Sample Simulation Waveform

Note: Testbench also includes a data comparator/checker. Completed data check indicates the correctness of the test. The data checker is bypassed when parameter *CUSTOM_INPUT* is set to *1* and custom image is used as an input.

6. In the transcript, the simulation result is shown like in Figure 4.14. The generated output RGB text image files are stored in the *source/testbench/test_vectors/* directory inside the project.

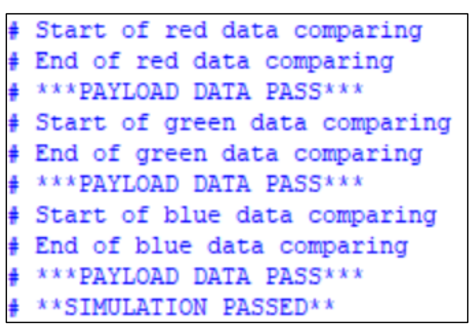
```
# Start of red data comparing
# End of red data comparing
# ***PAYLOAD DATA PASS***
# Start of green data comparing
# End of green data comparing
# ***PAYLOAD DATA PASS***
# Start of blue data comparing
# End of blue data comparing
# ***PAYLOAD DATA PASS***
# **SIMULATION PASSED**
```

Figure 4.14. Data Check Passed

7. The output RGB text files is used to generate an equivalent image in PNG format using the *txt_to_png.py* script present in the folder *source/testbench/test_vectors/txt_to_png.py* for visual inspection. To run the python script, open the

command prompt in the *source/testbench/test_vectors/* directory and enter *python txt_to_png.py*. After the script is executed, an output image named *HDMI_out.png* is generated in the *test_vectors* folder and *Images generated successfully!* will be printed on the screen as shown in Figure 4.15. In addition to the output image, if *CUSTOM_INPUT* is set to *0* the script also converts the reference RGB files into PNG format named as *golden_image.png*. No additional reference image is generated if *CUSTOM_INPUT* is set to *1*.

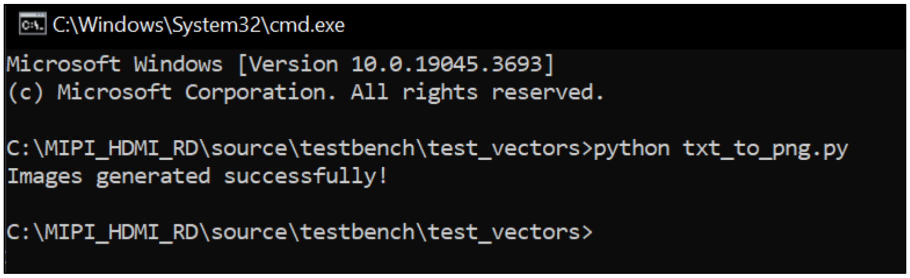

Figure 4.15. Running txt_to_img.py Script

8. The full frame simulation with checker may take a long time to complete. To study the behavior of the design without any checker, proceed to steps 9–11.
9. Set the Debayer IP Vertical pixel size in the test parameters to the minimum and match it to the Video Frame Buffer IP video frame height configuration as shown in Figure 4.16.

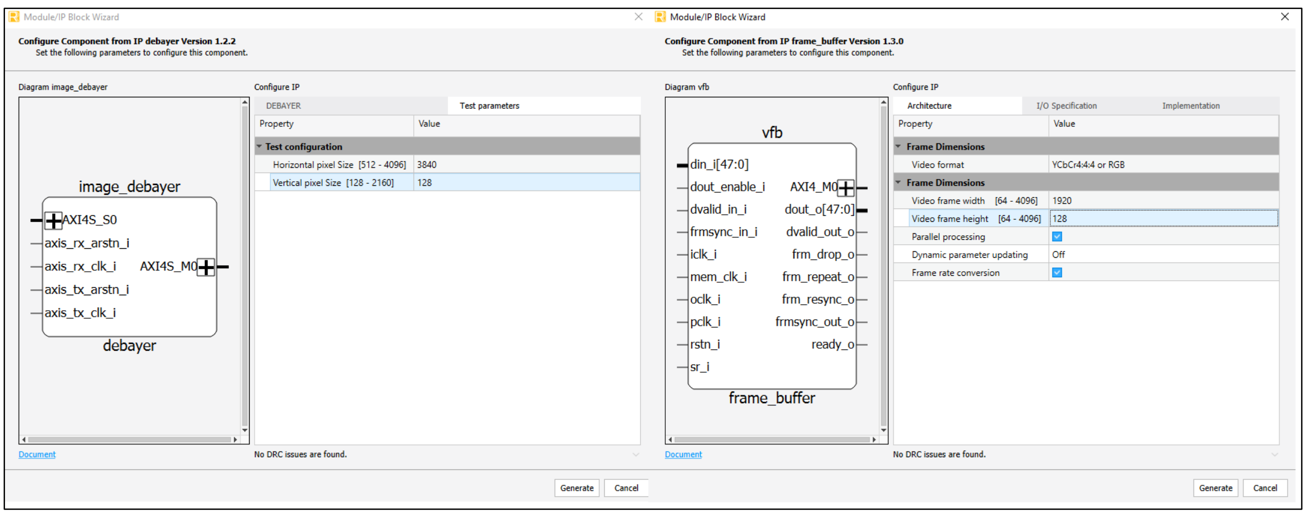

Figure 4.16. IP Modification to Speed Up Simulation Time

10. In the source/testbench/tb_top.sv file, modify VACTIVE to match values from step 9, and reduce the blanking periods to the minimum value. Figure 4.17 shows an example modification to the file.

```
//User editable parameters
parameter CUSTOM_INPUT = 0;  //Parameter to choose source bayer file, if
parameter DEBUG = "ON";  //Parameter to debug simulation; prints output

parameter NUM_FRAMES = 2;  //Do not change for now
//Local parameters required for simulation
//4800 WORD_CNT (Word Count) corresponds to 4K simulation whereas 2400 W
localparam RESOLUTION = (WORD_CNT == 4800) ? 1 : 0;   //if RESOLUTION is
localparam H_ACTIVE = (RESOLUTION == 1) ? 3840 : 1920;
localparam V_ACTIVE = (RESOLUTION == 1) ? 128 : 128;
localparam H_FRONT_PORCH = (RESOLUTION == 1) ? 10 : 10;
localparam H_SYNCH = (RESOLUTION == 1) ? 10 : 10;
localparam H_BACK_PORCH = (RESOLUTION == 1) ? 10 : 10;
localparam V_FRONT_PORCH = (RESOLUTION == 1) ? 3 : 3;
localparam V_SYNCH = (RESOLUTION == 1) ? 5 : 5;
localparam V_BACK_PORCH = (RESOLUTION == 1) ? 10 : 10;
localparam H_SYNC_BEGIN = (RESOLUTION == 1) ? 4020 : 2013;
localparam V_SYNC_BEGIN = (RESOLUTION == 1) ? 2168 : 1083;
localparam H_SYNC_END = (RESOLUTION == 1) ? 4108 : 2057;
localparam V_SYNC_END = (RESOLUTION == 1) ? 2178 : 1088;
localparam FIFOS_DEPTH = (RESOLUTION == 1) ? 8192 : 4096;
localparam PPC = NUM_TX_CH_INPUT;
```

Figure 4.17. Testbench Parameters Modification to Speed Up Simulation Time

11. Re-run the simulation steps 1–5. Note that the simulator gives errors due to mismatch on the expected data.

4.3. Implementation and Demonstration

Use the Radiant software to generate the design bit file after all the IPs have been configured as described depending on the operating resolution in the IP Descriptions section. Use the Lattice Programmer software to program the evaluation board using the generated bit file. This section describes the process of programming the evaluation board.

4.3.1. Generating the Bit File

Once the project is open, click on the green button or *Export Files* icon to start the synthesis and generation of .bit file as shown in Figure 4.18.

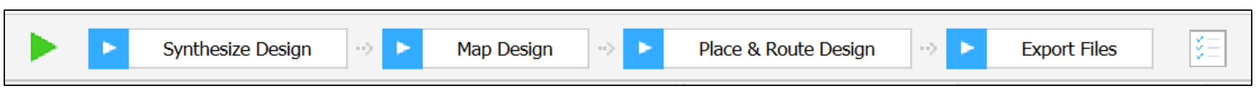

Figure 4.18. Exporting .bit File

- To successfully generate a bitstream, you must have installed licenses for the Lattice IPs used in the design. If you do not have a license for one or more IPs, use the “IP Evaluation” feature available in Radiant Software as shown in Figure 4.19 to create a time limited version of the bitstream.

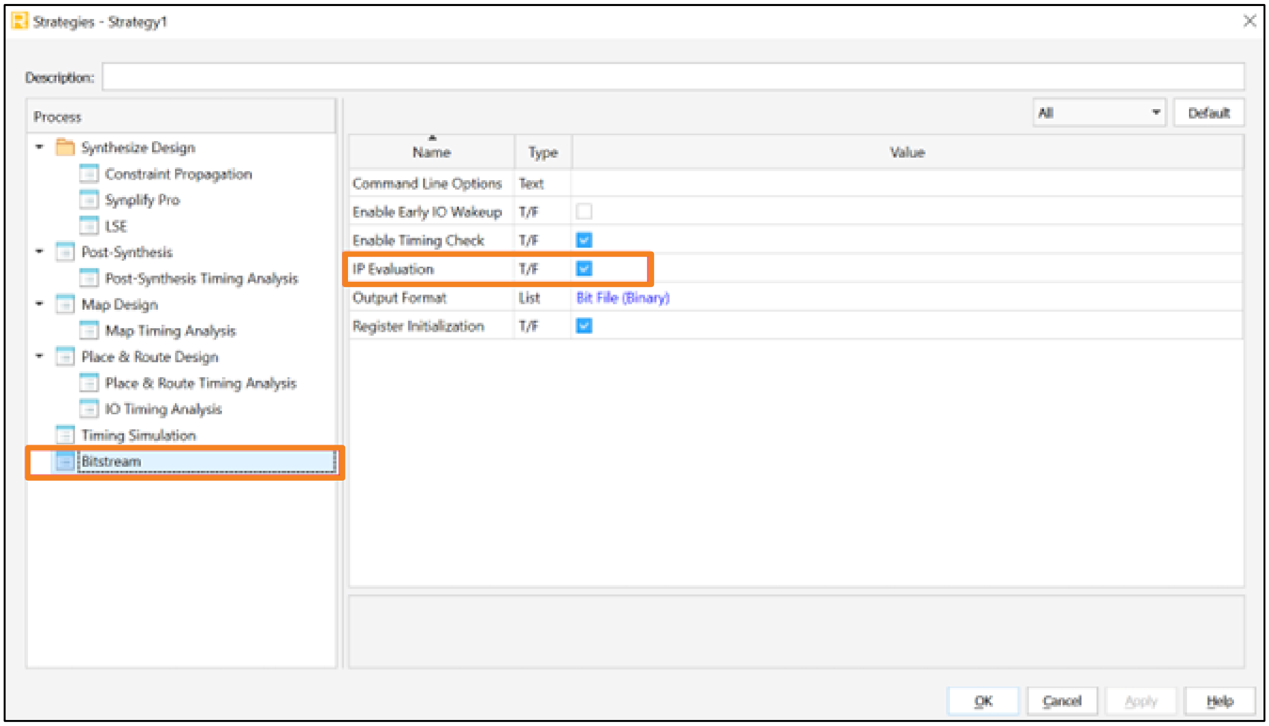

Figure 4.19. IP Evaluation

4.3.2. Uploading the .bit File and Programming the Board

This section describes the procedure to program the evaluation board.

1. Open the Lattice programmer from the Radiant software top toolbar menu as shown in Figure 4.20.

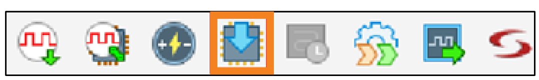

Figure 4.20. Programmer Icon

2. Use the following settings in the Programmer:
   - Device Family: Avant-AT_ENG
   - Device: LAV-AT-E70ES1
   - Operation: Fast Configuration
   - Programming Speed Settings (under cable setup): Use default clock divider
   - I/O Settings (under cable setup): Use default I/O settings
3. Select the generated bit file inside the implementation as shown in Figure 4.21.

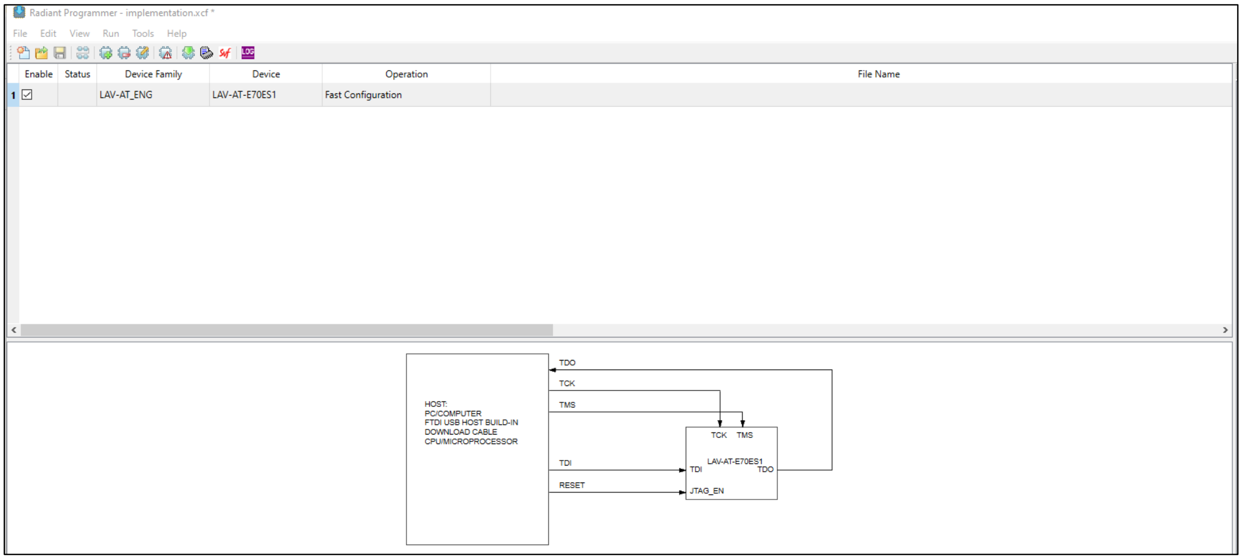

Figure 4.21. Lattice Programmer Settings

4. Click on the *Program Device* icon from the programmer toolbar menu as shown in Figure 4.22.

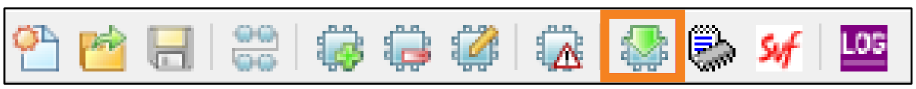

Figure 4.22. Program Device

5. Limitations

- For 4K at 30 Hz, the required pixel clock is 297 MHz according to the CTA-861 specification. In this demonstration, the LVCMOS18 I/O type is used to drive the HDMI outputs to the daughter card. The LVCMOS18 I/O type maximum frequency is only 250 MHz according to the Lattice Avant Platform - Specifications Data Sheet (FPGA-DS-02112).
- A full frame simulation with checker may take a long time to complete. You can shorten the simulation time by changing the VACTIVE parameter and run the simulation without any checkers.

Appendix A. Resource Utilization

Table A.1 shows the resource utilization of the CSI-2 to HDMI demonstration design for 4K and 1080p resolutions on the LAV-AT-E70ES1-3LFG1156C device using the Synplify Pro synthesis tool.

Table A.1. CSI-2 to HDMI Design Resource Utilization

Resolution	Pixels per clock	Color Format	Registers	LUTs	EBRs
4K	2	RGB	17449	20624	60
1080p	2	RGB	17397	20581	59

References

- CSI-2/DSI D-PHY Rx IP User Guide (FPGA-IPUG-02081)
- Byte-to-Pixel Converter IP User Guide (FPGA-IPUG-02079)
- Debayer IP User Guide (FPGA-IPUG-02203)
- Automatic White Balance IP User Guide (FPGA-IPUG-02204)
- Color Correction Matrix IP User Guide (FPGA-IPUG-02214)
- PLL Module User Guide (FPGA-IPUG-02220)
- FIFO Memory Modules User Guide (FPGA-IPUG-02182)
- EBR Memory Modules User Guide (FPGA-IPUG-02190)
- Lattice Avant Platform – Specifications Data Sheet (FPGA-DS-02112)
- Avant-E web page
- Lattice Radiant Software web page
- Lattice Solutions IP Cores web page
- Lattice Solutions Reference Designs web page
- Lattice Insights web page for Lattice Semiconductor training courses and learning plans

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

Revision History

Revision 1.2, March 2025

Section	Change Summary
Introduction	Updated the reference to the *Lattice Radiant Software user guide* in Table 1.1. CSI-2 to HDMI Reference Design Summary.
Opening and Running the Project	Updated the reference to the *Lattice Radiant Software user guide*.
References	• Removed *Lattice Radiant Software User Guide*. • Added *Lattice Solutions IP Cores* and *Lattice Solutions Reference Designs* web pages.

Revision 1.1, December 2024

Section	Change Summary
All	Made editorial fixes.
Acronyms in This Document	Added *LED*.
Introduction	• Updated *Targeted Devices* and *Lattice Implementation* in Table 1.1. CSI-2 to HDMI Reference Design. • Updated the requirements in the Hardware and Software sections. • Updated the descriptions in the Supported Configurations, LED and Push Button Description, • Jumper Settings, and Demonstration Hardware Setup sections. • Updated Table 1.2. LED Description, Table 1.4. Board Jumper Settings, and Figure 1.1. Demonstration Hardware Setup.
Functional Descriptions	• Updated the descriptions in the CSI-2 to HDMI Core Design, Simulation Testbench, and System Clocks PLL sections. • Updated Figure 2.1. CSI-2 to HDMI Core Block Diagram and Figure 2.3. CSI-2 to HDMI Demonstration Design Block Diagram. • Removed the *Pixel Clocks PLL* and *Frame Buffer* sections and updated the heading numbers of remaining sections accordingly.

IP and Signal Descriptions	• In the IP Descriptions section:
	  • Removed the *FIFO DC* IP and added *Frame Buffer* and *LPDDR4 Memory Controller* IPs to the description.
	  • Removed the following design consideration: *Debayer, AWB, and CCM IPs must use a common clock with a frequency slightly higher than the B2P's operating frequency*.
	• Updated Figure 3.1. CSI-2 to HDMI Demonstration Design Block Diagram, Figure 3.2. Module/IP Block Wizard of Byte-to-Pixel Converter IP, and Figure 3.3. Module/IP Block Wizard of Debayer IP.
	• In Table 3.1. MIPI CSI-2 D-PHY IP Attributes:
	  • Updated the *Rx Line Rate*, *Sync Clock*, and *Misc Signal*s attributes.
	  • Removed the *Type*, *Implementation*, *Depth*, and *Packet Delay* attributes.
	• Updated the *Enable AXI-4 Stream Tx Interface* attribute in Table 3.2. Byte-to-Pixel Converter IP Attributes.
	• Removed the *Pixel Clocks PLL IP* and *FIFO DC IP for Output Buffer* sections.
	• Added the Video Frame Buffer IP and LPDDR4 Memory Controller IP sections and updated the heading numbers of remaining sections accordingly.
	• Updated the descriptions in the System Clocks PLL IP section.
	• Updated Figure 3.8. Module/IP Block Wizard of System Clocks PLL IP.
	• Updated Table 3.8. System Clocks PLL IP Attributes.
	• In Table 3.13. CSI-2 to HDMI Core Design Signals:
	  • Updated the *clk_pixel_debayer_i* port.
	  • Removed the *clk_frame_buf_ref_i* port.
	  • Added the *reset_hdmi_n_i*, *clk_mcddr_ref_i*, *pclk_i*, *prstn_i*, *mc_train_err_o*, *mc_init_done_o*, *b2p_deb_line_err_o*, and *b2p_deb_frame_err_o* ports.
	• In Table 3.14. Demonstration Design Signals:

Section	Change Summary
	• Updated the *clk_i* port. • Removed the cam_i2c_config_done_o and hdmi_i2c_config_done_o ports. • Added the *mcddr_refclk_i* and *dbg_led_o* ports.

Opening and Running the Project	• Added step 3 and updated step 4 in the Opening the Project section.
	• In the Testbench Setup for Simulation section:
	  • Removed the previous step 3, which is to comment out the *FAMILY*, *RX_TYPE*, *NUM_RX_LANE*, *RX_GEAR*, *LAV_AT*, *ap6a00b*, and *LAV_AT_500E* parameter.
	  • Replaced the *Automatic White Balance IP* information with the *Video Frame Buffer IP* in the *things to consider* list.
	• In the Running Functional Simulation section:
	  • Updated Figure 4.10. Simulation Wizard and its caption.
	  • Updated step 3.
	  • Updated Figure 4.12. Run Simulation Value of 0 to Run All Simulations.
	  • Added steps 8–11.
	• Removed the *66292000 ERROR* and *small hold timing violations* descriptions in the Generating the Bit File section.
	• In the Uploading the .bit File and Programming the Board section:
	  • Updated the *Device Family* and *Device* settings in step 2.
	  • Updated Figure 4.21. Lattice Programmer Settings.
	  • Removed the previous step 5, which is on the checking firmware output window.

Limitations	Added this section.
Resource Utilization	• Updated the device in this section. • Updated Table A.1. CSI-2 to HDMI Design Resource Utilization.
References	Added the *Lattice Avant Platform - Specifications Data Sheet (FPGA-DS-02112)*.

Revision 1.0, December 2023

Section	Change Summary
All	Initial release.

www.latticesemi.com